UMEÅ UNIVERSITY

Bridging AI and Privacy: Solutions for High-Dimensional Data and Foundation Models

Sonakshi Garg

Doctoral Thesis, May 2025
Department of Computing Science
Umeå University
Sweden

Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

sgarg@cs.umu.se

Copyright © 2025 by Sonakshi Garg
Except Illustrations and Tables from
Paper I, © SCITEPRESS, 2023 (CC BY–NC–ND 4.0)
Paper II, © The authors published by Elsevier, 2024
Paper III, © IFIP, published by Springer Nature Switzerland AG, 2024
Paper IV, © The authors under exclusive license to Springer Nature Switzerland AG, 2025
Paper VI, © The authors under exclusive license to Springer Nature Switzerland AG, 2024

ISBN 978-91-8070-686-5 (print)
978-91-8070-687-2 (digital)
ISSN 0348-0542
UMINF 25.08
Cover illustrated by Ida Åberg

Printed by Scandinavian Print Group, Umeå, 2025

Abstract

The widespread adoption of machine learning (ML) in various domains has enabled the extraction of meaningful insights from complex, large–scale datasets. However, recent research has revealed that ML models are vulnerable to a range of privacy attacks which can expose sensitive information about the individuals in the training data. With regulatory frameworks like the General Data Protection Regulation (GDPR) which enforces strict requirements on data sharing, the need for privacy–preserving solutions has become increasingly critical. As the world becomes more digital, massive volumes of data are generated, often in high–dimensional spaces, where the number of attributes matches or exceeds the number of samples. ML models are extensively used to process such data, making it critical to protect both the data and the models from privacy attacks.

Traditional anonymization techniques such as k–anonymity and differential privacy often fall short when applied to high–dimensional datasets, because as dimensionality of the data increase, data–points tends to concentrate in the sparse regions of the feature space, making it difficult to find clusters of similar records. Therefore, this thesis proposes a set of privacy–preserving methodologies tailored for high–dimensional data and large–scale foundation models.

In this thesis, we begin by exploring manifold learning techniques to project high–dimensional data into a lower-dimensional latent space while preserving the intrinsic geometric structure of the original data. This transformation enhances the effectiveness of anonymization while maintaining data utility. Building on this, we then present a novel hybrid privacy method that integrates the strengths of k–anonymity with differential privacy, enabling robust anonymization that preserves both privacy and the underlying data structure. We further investigate synthetic data generation as a privacy–preserving alternative to using sensitive data, leveraging advanced generative models such as GANs and VAEs to produce high–quality synthetic datasets. To enhance the quality of the generated data, we propose techniques that preserve the intrinsic structure of the original high–dimensional data and incorporate prior domain knowledge to guide the generation process. We rigorously evaluate the synthetic data in terms of statistical fidelity, privacy risks, ML utility, and distributional capabilities through detailed visualizations. We then address high–dimensionality and privacy concerns in the context of large–scale foundation models. We propose

two model compression strategies using knowledge distillation and pruning, that effectively reduce the number of model parameters while preserving performance and enhancing the privacy of the system.

Collectively, the thesis contributes towards building privacy–aware AI systems by developing practical solutions that address the complex interplay between high–dimensionality and privacy models.

Sammanfattning

Den utbredda användningen av maskininlärning (ML) inom olika områden har gjort det lättare att utvinna meningsfulla insikter ur komplexa, storskaliga datamängder. Ny forskning har dock visat att ML-modeller är sårbara för en rad integritetsattacker som kan avslöja känslig information om enskilda personer i träningsdata. Med regelverk som General Data Protection Regulation (GDPR), som ställer strikta krav på datadelning, har behovet av integritetsskyddande lösningar blivit allt viktigare. I takt med att världen blir alltmer digital genereras enorma mängder data, ofta i högdimensionella, där antalet attribut stämmer överens med eller överstiger antalet datapunkter. ML-modeller används i stor utsträckning för att bearbeta sådana data, vilket gör det viktigt att skydda både data och modeller från integritetsattacker.

Traditionella anonymiseringstekniker som k-anonymitet och differentiell integritet kommer ofta till korta när de tillämpas på högdimensionella datamängder, eftersom datapunkter tenderar att koncentreras i glesa regionerna inom omfånget av egenskaper när datadimensionaliteten ökar, vilket gör det svårt att hitta kluster med liknande poster. Därför föreslår denna avhandling en uppsättning integritetsskyddande metoder som är skräddarsydda för högdimensionella data och storskaliga grundmodeller.

I den här avhandlingen börjar vi med att utforska tekniker för mångfaldig inlärning för att projicera högdimensionella data i ett lägre dimensionellt latent omfång samtidigt som vi bevarar den inneboende geometriska strukturen i originaldata. Denna omvandling förbättrar anonymiseringens effektivitet samtidigt som dataanvändbarheten bibehålls. På grundval av detta presenterar vi sedan en ny hybridintegritetsmodell som integrerar styrkorna hos k-anonymitet med differentiell integritet, vilket möjliggör robust anonymisering som bevarar både integritet och den underliggande datastrukturen. Vi undersöker vidare generering av syntetiska data som ett integritetsbevarande alternativ till att använda känsliga data, och utnyttjar avancerade generativa modeller som GAN och VAE för att producera syntetiska datamängder av hög kvalitet. För att förbättra kvaliteten av genererad data föreslår vi tekniker som bevarar den inneboende strukturen i de ursprungliga högdimensionella datan och införlivar tidigare domänkunskap för att vägleda genereringsprocessen. Vi utvärderar de syntetiska uppgifterna noggrant med avseende på statistisk tillförlitlighet, integritetsrisker, ML-värde och distributionsegenskaper genom detaljerade vi-

sualiseringar. Vi tar sedan itu med hög dimensionalitet och integritetsfrågor i relation till storskaliga grundmodeller. Vi föreslår två modellkomprimeringsstrategier med hjälp av kunskapsdestillation och beskärning, som effektivt minskar antalet modellparametrar samtidigt som prestanda bevaras och systemets integritet förbättras.

Sammantaget bidrar avhandlingen till att bygga integritetsmedvetna AI-system genom att utveckla praktiska lösningar som hanterar det komplexa samspelet mellan hög dimensionalitet och integritetsmodeller.

Preface

This thesis presents the development of privacy–aware AI techniques by integrating various privacy models, manifold learning methods, and model compression strategies. This work is based on the following research papers.

Paper I **Sonakshi Garg** and Vicenç Torra. K–Anonymous Privacy Preserving Manifold Learning. *International Conference on Security and Cryptography (SECRYPT)*, pp. 37-48. SciTePress, 2023

Paper II **Sonakshi Garg** and Vicenç Torra. Privacy in manifolds: Combining k–anonymity with differential privacy on Fréchet means. *Computers and Security*, 103983. Elsevier, 2024

Paper III **Sonakshi Garg** and Vicenç Torra. Can Synthetic Data preserve manifold properties? *IFIP International Conference on ICT Systems Security and Privacy Protection (IFIPSEC)*, pp. 134-147. Cham: Springer Nature Switzerland, 2024

Paper IV **Sonakshi Garg** and Vicenç Torra. Exploring Distribution Learning of Synthetic Data Generators for Manifolds. *European Symposium on Research in Computer Security*, pp. 65-76. Cham: Springer Nature Switzerland, 2025

Paper V **Sonakshi Garg**, Marcel Neunhoeffer, Jörg Drechsler and Vicenç Torra. Using Prior Knowledge to Improve GANs for Tabular Data Without Compromising Privacy. *Submitted.*

Paper VI **Sonakshi Garg** and Vicenç Torra. Task–Specific Knowledge Distillation with Differential Privacy in LLMs. *European Symposium on Research in Computer Security (ESORICS)*, pp. 374-389. Cham: Springer Nature Switzerland, 2024

Paper VII **Sonakshi Garg** and Vicenç Torra. PrunePrivyTune: Accelerating Language Models with Pruning and Differentially Private Fine-Tuning. *Submitted.*

In addition to the papers included in this thesis, the following publication was published within the studies but not included in this doctoral thesis.

Paper VIII **Sonakshi Garg**, Hugo Jönsson, Gustav Kalander, Axel Nilsson, Bhhaanu Pirange, Viktor Valadi and Johan Östman Poisoning Attacks on Federated Learning for Autonomous Driving. *Scandinavian Conference on Artificial Intelligence, pp. 11-18*, 2024.

This study was partially funded by the Wallenberg AI, Autonomous Systems and Software Program (WASP), funded by the Knut and Alice Wallenberg Foundation.

Dedicated to my family for their constant love and support

Acknowledgements

It's often said that a PhD is more about the journey than the degree itself–and looking back, I truly believe that to be true. This journey, with all its highs and lows, has shaped me just as much as the final destination. Leaving home and moving abroad for the first time in February 2022 was not only a big step, but a deeply personal one–filled with moments of uncertainty, growth, and self–discovery. I feel truly grateful for the incredible people who stood by me, lifted me up, and made this path feel a little less lonely and a lot more meaningful.

First and foremost, I am deeply grateful to **Vicenç** for your unwavering support, insightful guidance, and constant encouragement throughout my PhD journey. Your belief in me, even during the moments I doubted myself, has meant more than words can express. I feel truly fortunate and blessed to have you as my supervisor. In our culture, we say *Yatha guru, tatha shishya*–as is the teacher, so is the student and I hope to carry your sense of clarity, depth, and kindness with me.

I am also thankful to **Pranab** Sir and **Sandeep** Sir, my master's supervisors, for encouraging me to pursue a PhD abroad–something that had never even crossed my mind until then. Your belief in my potential gave me the courage to take this leap. A heartfelt thanks especially to Pranab Sir, who not only guided me academically but also helped reassure my parents when they were unsure about me moving so far from home.

I'd like to thank **Zoe Falomir**, both a mentor and a friend, for encouraging me to explore new opportunities beyond my PhD research. Your presence has been a constant source of motivation and refreshment.

A special thanks to **Mariam**–from the moment we met, I felt an instant bond with you. Your calm, thoughtful nature and mature approach to life have had a profound influence on me. I've learned so much from your steady presence–how to stay composed under pressure, think rationally, and approach challenges with patience and clarity. I'm truly grateful that we've shared this PhD journey, supporting each other through every high and low. I deeply value our friendship and sincerely hope it continues to grow and lasts a lifetime.

I will always cherish the time spent with **Ayush** and **Sargam** throughout this journey. From taking our WASP courses together–to exploring the beauty of Sweden, we created memories that I'll carry with me forever. Ayush's easy-

going nature, and the way he always brought laughter made every moment so enjoyable. Sargam, with her kindness, helpful spirit, and cheerful personality, made every interaction light and uplifting, I truly enjoyed all the time we shared. I'll forever treasure the moments spent with **Fatemeh**–experiencing the warmth of her generous heart and unwavering care. Her presence has been a constant source of comfort and support throughout this journey. A heartfelt thanks to **Zuzana**, for being a guiding light during the PhD path. Your wisdom, encouragement and openness in sharing your experiences as a postdoc have meant so much. To **Jed**, as you embark on your PhD journey–I wish you all the very best. You bring a fresh perspective and great energy to the group. And to **Sudipta**, thank you for your helpful nature and willingness to support everyone with a smile. I'm deeply grateful to the entire NAUSICA group, including **Shekhar**, for being an essential part of this journey. The joyful lunches and meaningful conversations have made this experience so much more vibrant and memorable. All the time spent together has been refreshing, encouraging, and full of light–hearted joy. Thank you all for being such a wonderful part of this chapter in my life.

A special mention to my brother **Jonas**, who filled the space of a brother in my life when I needed it the most. I'm so grateful that I could celebrate Indian festivals with you–especially Rakshabandhan, which is closest to my heart. Our meditation and spiritual chanting sessions brought me so much peace, especially during the times I felt low. Now, I can proudly say that I have two brothers–and I truly hope our bond stays this strong, always.

I feel so lucky to have met **Sushma** in my very first Swedish class. Even though our initial experience at SFI wasn't the best, I'm grateful because that's how I found a wonderful friend like you. From learning Swedish to our momo and golgappa evenings, and the warmth of cooking together–I'll always cherish those cozy moments. Thank you for your love and laughter.

To **Petra**–thank you for introducing me to Swedish traditions, teaching me to bake those lovely Swedish desserts, trying out different activities together, and most of all, for caring for me like a mother. I'll always hold deep love and respect for you.

I'm forever grateful to **Anshul**–for your constant support, for always being there to listen, and simply for your presence. It has meant so much to me.

To **Priya**, who has stood by me through thick and thin–even from miles away. No distance could ever dim the light of your friendship, which has been my anchor through it all. Cheers to us, and to a bond that only grows stronger with time.

To my *dearest parents* no words could ever fully express the love and gratitude I hold in my heart for you. This journey, with all its challenges and triumphs, would not have been possible without your unwavering belief in me. You've been my constant pillars. Every call, every message, every silent prayer from you gave me the strength to keep going. Thank you for giving me the freedom to dream, the courage to pursue those dreams, and the emotional anchor to return to whenever I felt lost. To my brother **Harsh**, thank you for

your quiet care and for always finding a way to bring a smile to my face–even if it was by teasing me endlessly. Your presence added moments of laughter and lightness, even during the most difficult times. This thesis is dedicated to my family–my heart, my home and my greatest source of strength. I am endlessly blessed to have you as my family.

Above all, I bow in deep gratitude to *Lord Radha Krishna* and *Lord Narasimha* for being my eternal source of strength and peace. I pray that my faith in you remains unwavering, and that your blessings continue to guide me through every step of life.

Sonakshi Garg

Contents

Chapter 1

Introduction

> The choice of problems is the primary determinant of what one accomplishes in science
>
> — *John Hopfield*

1.1 Importance of Privacy in the Digital Era

In today's digital age, personal data has become a highly valuable asset. Every online interaction–whether browsing the web, using mobile applications, or engaging with smart devices generates vast amounts of data. While businesses and organizations use this data to improve services and decision–making, its collection often occurs without individuals' full awareness or control, raising serious privacy concerns. The rapid advancement of artificial intelligence (AI) has further intensified these concerns. AI systems analyze large datasets to automate decisions, personalize experiences, and enhance efficiency. From generative AI that creates content based on user inputs to smart assistants that learn personal preferences, AI relies heavily on personal data. While these technologies offer convenience, they also introduce risks, including unauthorized surveillance, data misuse, and identity theft. A common example is targeted advertising, seeing an ad for a product moments after discussing it with a friend or receiving health–related recommendations based on the recent purchases. Such instances reveal how companies track and analyze personal data, often without explicit consent. The key issue is not just personalization but also the lack of transparency and control over how data is collected, shared, and used.

Privacy is a fundamental human right, ensuring the individuals to control their personal information and its usage. The Universal Declaration of Human Rights states this in its Article 12, UN General Assembly.

> **Article 12.** No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation. Everyone has the right to the protection of the law against such interference or attacks [Uni48].

In Europe, the General Data Protection Regulation (GDPR) [GDP18] has been in force since 2018, establishing a comprehensive legal framework for data protection and privacy. It clearly defines personal data and data processing, outlining the responsibilities of organizations handling such information. GDPR grants individuals key rights, including the right to erasure and right to rectification, empowering them with greater control over their personal data. Additionally, it mandates timely reporting of data breaches and enforces strict penalties for non–compliance, promoting transparency, accountability, and stronger safeguards against data misuse.

Several countries have enacted regulations in their constitutions similar to GDPR such as in the United States, they have Health Insurance Portability and Accountability Act (HIPAA,1996) [HHS96] and California Consumer Privacy Act (CCPA) [Par18] etc. These regulations set the legal framework for how personal data should be handled, ensuring individuals' privacy rights are respected across various sectors.

The application of advanced data analysis techniques to personal data enables the discovery of behavioral patterns, facilitates trend prediction, and contributes to the optimization of personalized services. When the data is processed by data controllers who adhere to privacy standards, the risk of privacy violations can be mitigated. However, in many cases, data sharing among different stakeholders is essential. For instance, personal data may need to be shared with software development firms for system testing or with data analysts leveraging AI models to derive insights about the customers to improve service offerings. This sharing, while beneficial, introduces significant privacy risks, as it increases the potential for unauthorized access, misuse, and breaches of confidentiality. Addressing these risks requires robust privacy frameworks, secure data–sharing protocols, and clear guidelines for data handling across organizations. As technology continues to evolve, privacy must remain a priority. Achieving a balance between innovation and ethical data use requires transparent AI systems, robust privacy frameworks, and policies that empower individuals to protect their personal information.

1.2 Motivation

Data privacy is a critical concern, particularly when handling sensitive personal information. Privacy regulations mandate strict protections to prevent unauthorized access and misuse of data. Privacy–Preserving Data Publishing (PPDP) [Hun+12; VC04] provides methodologies to share valuable insights while ensuring individuals' privacy. A key challenge in PPDP is to ensure that anonymized data remains useful for downstream tasks while protecting

individuals' privacy. One common approach within PPDP is data anonymization, where raw data is transformed through techniques such as generalization, suppression and perturbation to minimize privacy risks.

However, traditional anonymization techniques often fall short, particularly for high–dimensional data. The real–world datasets originate from various resources such as online platforms, financial institutions, healthcare systems, and smart city applications, where large–scale data analysis fuels predictive modeling, AI–driven decision–making, and real–time analytics. Such modern datasets have number of attributes often exceeding the number of individuals in the dataset. Various organizations, including government agencies and healthcare institutions, collect and share such data such as census records and medical histories, with third parties for specific analytical purposes. However, directly releasing raw data may expose individuals to privacy risks. An adversary can exploit auxiliary information from external sources, such as voter lists, to re–identify individuals, undermining the intended privacy protections [De +12].

High–dimensional datasets introduce unique challenges for anonymization. As dimensionality increases, data points tend to concentrate in the sparse regions of the feature space, making it difficult to form sufficiently large groups of similar records. This sparsity reduces the effectiveness of traditional privacy–preserving mechanisms. Various privacy models such k–anonymity [Sam01], l–diversity [Mac+07], t–closeness [LLV06] and approaches such as [LGS13; TMK08; Zhu+17] have been proposed to address privacy concerns. However, the current mechanisms for these privacy models diminishes in high–dimensional settings due to the difficulty of finding meaningful equivalence classes. Even implementation of an alternative privacy model, Differential Privacy (DP) [Dwo06], requires injecting too much noise to ensure privacy guarantees, which can severely degrade the usability of the data. Unfortunately, as the dimensionality increases, most existing privacy techniques struggle to handle high–dimensional data effectively [Agg05; Agg06; GTK08] due to two fundamental limitations.

First, privacy preservation in high–dimensional data often results in severe utility degradation [Fun+11]. With the increase in number of attributes, adversaries have more information to compare with external data, making it easier to identify individuals. To counter this, stronger perturbation is required, leading to a significant loss of data utility. Second, the spatial locality assumption used in many anonymization techniques, such as generalization–based methods [Bre+14; HN09; LGS13], becomes impractical in high–dimensional spaces. This sparsity leads to greater distances between points, making it difficult to find clusters of similar records. As a result, traditional anonymization techniques become ineffective [Agg06].

Some techniques have been proposed to address the problem of dimensionality in privacy models. Feature selection, feature transformation, and partitioning techniques have been widely explored to reduce dimensionality while preserving privacy [Var+12; CS14; Li+19; RR20; Wan+20a]. Principal Component Analysis (PCA) [AW10] is also commonly used to project database onto

lower–dimensional representations while retaining key characteristics, and then privacy models could be used on the low–dimensional database. However, these methods primarily work well for linear data, but falls short when dealing with non–linear structure of the data [SSM98], necessitating advanced techniques capable of handling non–linearly distributed high–dimensional data while ensuring privacy. Given these challenges, there is a need for novel privacy–preserving mechanisms tailored for high–dimensional data. A promising direction is the integration of manifold learning techniques [TSL00] with privacy models to capture intrinsic data structures in a lower–dimensional space, which works well for non–linear data structures while maintaining privacy guarantees.

Furthermore, personal data is frequently utilized by machine learning (ML) models for prediction, decision–making, and analytics. While anonymization aims to protect raw data, ML models and aggregated outputs can also pose privacy risks. These models may retain implicit patterns from the training data, enabling adversaries to infer sensitive information through attacks such as membership inference [Sho+17] and model inversion [FJR15]. Also, ML models frequently encounter these challenges when dealing with such high–dimensional data [JT09]. This highlights the need for Privacy Preserving Machine Learning (PPML) techniques that ensure data protection at different stages: before training (on input data), during training the model (privacy on computation), or after training (on the output). The choice of a privacy model depends on the specific application and the required level of protection. For instance, differential privacy provides a formal framework to limit the influence of any single data point on the model, ensuring privacy–preserving computations.

In addition to employing PPDP and PPML techniques to protect sensitive data, an alternative approach is to generate synthetic data that closely mimics the statistical properties and patterns of the original dataset [Rub93; Dre11]. By replacing real data with synthetically generated data, privacy risks can be mitigated, as no actual personal information is shared, published, or used for model training and analysis. However, several challenges arise with this approach. First, generating high–quality synthetic data that accurately preserves the distributions and relationships of high–dimensional real–world data is non–trivial. Poorly generated synthetic data can fail to capture the necessary statistical dependencies, reducing its usefulness. Second, ensuring that synthetic data is truly privacy–preserving is crucial, as merely generating synthetic data does not guarantee privacy unless rigorous privacy assessments are conducted. There is a risk that synthetic data may still leak sensitive patterns or allow attackers to infer information about individuals from the original dataset [HAP17; Sho+17]. Third, improving the performance of synthetic data generators, such as Generative Adversarial Networks (GANs) [Goo+20] or Variational Autoencoders (VAEs) [Kin13], remains an ongoing research challenge, particularly in balancing privacy and utility.

Further, high–dimensionality is not only a concern for data but also for modern machine learning models. Many large–scale models, such as deep learning architectures and NLP models, contain millions to billions of param-

eters, leading to high computational costs and latency. The inference time of a model that is the time taken to respond to a query must be in the order of milliseconds for real–world deployment. These models often learn from high–dimensional data, also increasing the significant computational overhead. As the models grow larger to achieve higher performance, their inference latency and resource demands also increase significantly, making them impractical for real–time applications. Several model compression techniques have been introduced in the literature to mitigate these challenges, including knowledge distillation [HVD15], pruning [ZG17], quantization [Wu+16; Gon+14], low–rank factorization [Che+05], and batch inference [CKY23]. These methods aim to reduce model complexity, but often comes at the cost of reduced model utility.

Beyond computational efficiency, large–scale models also pose critical privacy and security risks. Deep neural networks, particularly transformer–based models, are susceptible to several privacy attacks [Sho+17; Car+21], where adversaries can reconstruct the sensitive training data or determine whether a specific individual was included in the training dataset. Techniques like differentially private training, have been explored to mitigate these risks, however with carefully designed attacks, it is still possible to recover certain information from the model. To address both scalability and privacy concerns, there is a strong need for research into privacy–preserving model compression techniques, which optimize models for efficiency while ensuring privacy guarantees.

The overarching goal of this thesis is to explore privacy–aware AI systems by investigating techniques that ensure data protection while maintaining utility in machine learning models and data publishing. Specifically, it will focus on enhancing privacy–preserving mechanisms in high–dimensional data, developing privacy–aware synthetic data generation methods, and establishing privacy frameworks for large–scale AI models. By systematically evaluating their effectiveness in terms of utility retention and privacy protection, this research contributes to the development of privacy frameworks that enable secure and robust AI systems.

1.3 Research Questions

Ensuring data protection for high–dimensional data is inherently challenging due to the trade–off between privacy and utility. The primary aim of this research is to develop privacy–aware AI systems that effectively protect high–dimensional data while preserving its usability for downstream tasks. Thus, the objective is to explore existing privacy models and manifold learning techniques. Existing mechanisms of privacy–preserving models, such as k–anonymity and differential privacy, struggle to maintain utility of anonymized datasets especially as dimensionality increases. Manifold learning offers a potential solution by leveraging the manifold hypothesis, which states that real–world high–dimensional data often lie on a lower–dimensional manifold

that is embedded in a high–dimensional space [Cay+08]. By leveraging the manifold structure of high–dimensional data, we obtain a lower–dimensional representation that preserves its intrinsic characteristics. Privacy models can then be applied more effectively to this low–dimensional data, reducing the need for excessive perturbation while maintaining data utility, which is a major drawback in high–dimensional data anonymization. Thus, we explore how manifold learning can be effectively utilized to protect high–dimensional data while maintaining utility. Additionally, we investigate whether hybrid privacy models, combining k–anonymity and differential privacy could be used instead of individual privacy models.

Furthermore, an alternative to direct anonymization is using high–quality synthetic data as a privacy–preserving mechanism. The key idea behind synthetic data generation is that if real data samples are not explicitly reproduced in synthetic datasets, they should theoretically be protected against adversarial attacks. We investigate whether high–fidelity synthetic data can be generated for high–dimensional datasets while preserving the statistical properties of the original data. We also explore techniques to enhance the performance of GANs and VAEs for tabular data generation, particularly in datasets with many categorical variables. Additionally, we investigate the vulnerabilities of synthetic data generators to privacy attacks and develop privacy–preserving generative models to mitigate these risks.

High–dimensionality is not only about data, but also about model parameters. There are large–scale models, such as deep learning and NLP models, which contain millions or even billions of parameters. These models pose two major challenges: computational inefficiency and privacy. Large–scale models have high inference times, because they are trained on millions of parameters, making real–time deployment difficult, also large–scale models are vulnerable to several privacy attacks. We aim to design privacy–aware AI systems that minimize computational overhead while maintaining strong privacy guarantees. To achieve this, we investigate the integration of model compression techniques such as knowledge distillation and pruning with privacy–preserving mechanisms.

The main, and specific research questions of this thesis are as follows.

RQ1: Are existing privacy models and their combinations effective in preserving the privacy and utility of high–dimensional data?

RQ2: Can synthetic data generation methods capture and preserve the intrinsic manifold structure of high–dimensional data?

RQ3: Can large–scale models like language models leverage privacy models and model compression to ensure privacy and reduce computational overhead?

1.4 Main Contributions

To answer the research questions, this thesis contributes to the development of privacy–aware AI systems by systematically investigating and designing techniques that ensure data protection while preserving the utility and effectiveness of machine learning models and high–dimensional data publishing. This integration enhances the effectiveness of PPDP and PPML techniques by improving data utility, privacy, and robustness against privacy attacks. We now describe how each research question is addressed and outline the key contributions.

In order to address **RQ1**, we aim to evaluate the effectiveness of existing techniques for privacy models, such as k–anonymity and DP, in balancing privacy and utility for high–dimensional data. Additionally, we investigate the synergies between k-anonymity and differential privacy, and if k-anonymity can help to improve the utility of DP responses. There are two main outcomes of this research question.

- Limitations of existing mechanisms for k–anonymity and differential privacy models, in preserving the utility of high–dimensional data are analyzed. To address them, a manifold learning based privacy–preserving framework is proposed. This framework introduces geodesic distance as an alternative to Euclidean distance, capturing the intrinsic structure of the data more effectively. Additionally, manifold learning techniques are employed to project high–dimensional data onto a lower–dimensional space before applying anonymization. By preserving meaningful geometric relationships, this approach enhances privacy while minimizing information loss, thereby achieving a more favorable trade–off between privacy and utility compared to conventional models.
- Design a hybrid anonymization technique, integrating the strengths of both k–anonymity and DP. Additionally, a relationship between the privacy parameters k and ϵ is established in terms of its impact on data utility.

These results are presented in Paper I and II.

In order to answer **RQ2**, we explore the challenges faced by synthetic data generators in producing high–quality synthetic data, with a focus on ensuring that the generated data is safe from privacy risks. We also examines whether incorporating prior knowledge about the data can enhance the performance of these generators. There are three main outcomes of this research question.

- Design a framework for generating high–fidelity synthetic data that preserves both privacy and the inherent structure of the data. This framework evaluates privacy risks through data reconstruction attacks and assess the utility by analyzing statistical and machine learning performance metrics.

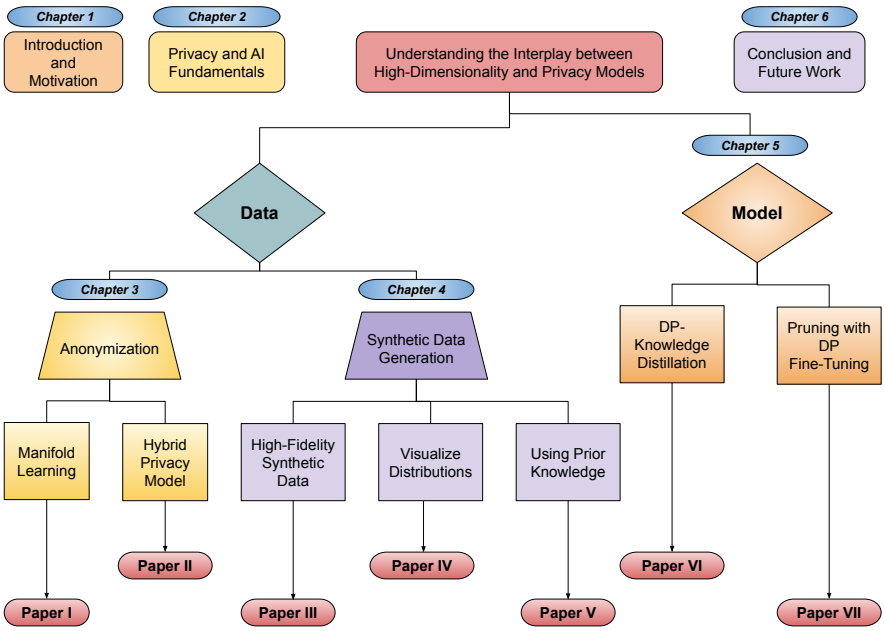


Figure 1.1: Thesis Outline

- Explore distributional learning capabilities of synthetic data generators, such as GANs and VAEs, to enhance their ability to capture and replicate the underlying data distribution effectively.

- Investigate the use of prior knowledge about the data to enhance the performance of GANs for tabular data, while ensuring that the privacy is not compromised.

These results are presented in Paper III, IV and V.

In order to address **RQ3**, we investigate the challenges of high–dimensionality in large–scale models, focusing on language models, particularly in terms of the number of parameters. We explore methods for compressing these models to reduce computational overhead while ensuring they remain secure against privacy attacks through the application of privacy model. There are two main outcomes of this research question.

- Design a task–specific knowledge distillation approach that combines transfer learning and differential privacy for model compression, ensuring the privacy of the model during the process.

- Develop a pruning strategy integrated with DP fine–tuning, ensuring that the pruning process is complemented by privacy protection, while evaluating privacy vulnerabilities through training data extraction attacks.

These results are presented in Paper VI and VII.
As a summary, our research focuses on understanding the interplay between high–dimensionality and privacy models and provides privacy–aware AI systems. Figure 1.1 provides an overview of the thesis outline, illustrating the relationships between research questions, key concepts, thesis chapters, and the corresponding papers.

1.5 Research Method

This research aims to enhance our understanding of privacy vulnerabilities in data publishing and machine learning while proposing viable privacy–aware AI solutions to mitigate these risks. These technological solutions can be considered as artifacts. To ensure a rigorous and systematic design process, we adopt the Design Science Research (DSR) methodology [Hev+04]. DSR focuses on the creation and evaluation of innovative IT artifacts that address well–defined problems, as depicted in Figure 1.2. The first step in DSR is problem identification, which involves recognizing a challenge that can be addressed through the development of an artifact. This is typically derived from a comprehensive literature review to establish the significance of the problem. In this thesis, we follow a similar approach, reviewing existing literature to identify the challenge posed by the interplay between data dimensionality and privacy models.

Following problem identification, the suggestion phase defines the research objectives and outlines the expected outcomes. The primary goal of this thesis is to develop privacy–aware AI solutions that balance privacy and utility. In the design phase, prototype solutions are developed iteratively. This stage involves refining ideas, designing models, and conducting preliminary experiments to evaluate feasibility. Next, in the demonstration phase, the refined solution is applied in relevant scenarios to assess its practical effectiveness. The evaluation phase follows, where the solution undergoes rigorous assessment based on predefined performance metrics, ensuring it meets privacy and utility requirements. Once the solution achieves the desired quality, it transitions to the conclusion phase, where final insights are drawn, and the findings are prepared for publication.

1.6 Thesis Organization

The thesis is organized as follows. Chapter 2 provides essential background concepts on various privacy models, techniques for handling high–dimensional data, and synthetic data generation methods, which forms the foundation for the subsequent chapters. Chapter 3 explores privacy–preserving solutions

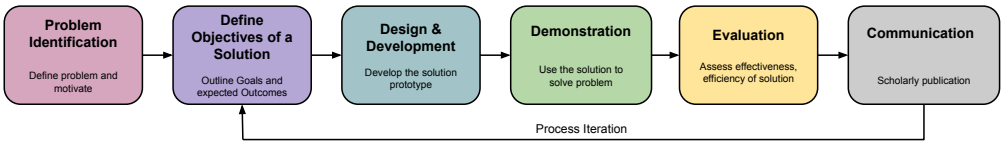


Figure 1.2: Design Science Research Methodology

specifically designed for high–dimensional data publishing. Chapter 4 presents synthetic data generation as an alternative to traditional anonymization techniques and evaluates its effectiveness. Chapter 5 discusses the challenges associated with large–scale models, such as language models, and proposes potential solutions. Chapter 6 concludes the thesis by summarizing key findings and outlining directions for future research.

Chapter 2

Privacy Models and Machine Learning

> Meet the first beginnings; look to the budding mischief before it has time to ripen to maturity
>
> — *William Shakespeare*

In today's data–driven world, vast amounts of data are collected and shared every second. This data is frequently analyzed using advanced statistical and data mining techniques to extract insights. When the data is analyzed internally within the organization that collected it, the risk of disclosing sensitive information remains relatively low. However, when the analysis requires collaboration with third parties, the risk of privacy breaches becomes significantly higher, making data privacy a critical concern. The concept of privacy originally emerged as a concern among statisticians, particularly in the context of publishing census data, where the objective was to prevent the disclosure of sensitive information. Over time, privacy concerns expanded into the field of computer science, addressing challenges related to data mining, secure computation, and data communication.

This chapter explores the multidimensional nature of data privacy, addressing key questions such as: whose privacy is being protected, how can data privacy be achieved during computations, how do privacy–preserving methods adapt to scenarios involving different numbers of data sources. We then introduce privacy–preserving methods [VC04; Hun+12], which includes Privacy Preserving Data Publishing (PPDP) and Privacy Preserving Machine Learning (PPML) that has developed comprehensive frameworks and mechanisms to ensure privacy protection for various types of data. These mechanisms encompass a wide range of anonymization techniques, categorized as perturbative (e.g., noise addition or data masking), non–perturbative (e.g., generalization

or suppression), and synthetic data generation methods. We then introduce different privacy models that formalize the definition of privacy. Finally, we delve into privacy considerations for machine learning applications, particularly when dealing with high–dimensional data. We discuss techniques such as manifold learning, which reduces the dimensionality while preserving the data's structure, and model compression, which optimizes computational efficiency without compromising privacy. These approaches enable effective data analysis while adhering to stringent privacy requirements.

2.1 Dimensions of Data Privacy

Data privacy is a discipline focused on developing theories, tools, and methodologies to ensure the proper governance of personal data. Different methods have been developed for different scenarios and under different assumptions on the data. According to [Dom07; Tor17], data privacy has been broadly categorized into three dimensions.

1. Whose privacy is being sought
2. The computations to be done
3. The number of data sources

The first dimension can be discussed considering a scenario which involves three actors.

- **Data subject/respondent**: This focuses on individuals that have generated the data i.e, customers, participants etc. We consider them as passive subjects as they cannot take actions to protect their own privacy.
- **Data controller/ holder**: This refers to an organization or individual who has gathered the data and owns the database. For instance service providers, government agencies etc.
- **Data user/recipient**: This is an authorized party responsible for interacting with the collected data. This includes activities such as visualizing, analyzing, or processing the data to derive insights or support decision-making.

The second dimension considers the prior knowledge that the data controller has about the usage of this data. This dimension emphasizes how the data will be utilized and the expected applications of the protected data. Data protection methods can be categorized based on their suitability for specific use cases, considering the level and type of prior knowledge available to the data controller. They can be categorized into three procedures.

- **Computation–driven or specific purpose protection procedure**: In this scenario, the analysis or computation to be performed on the data is known beforehand. Consequently, the protection procedures are specifically tailored to align with the intended purpose.

- **Data–driven or general purpose protection procedure**: In this scenario, no specific analysis is anticipated for the data. These procedures are designed to provide broad privacy protection, making the data suitable for diverse and unforeseen applications. A common example includes datasets published for public use by government agencies or research organizations.

- **Result–driven protection procedure**: In this scenario, the focus of privacy is on the outcomes generated from applying a specific data mining method to a particular dataset [Ata+99; Atz+08].

The third dimension corresponds to the number of data sources, which could be distinguished in two cases: (i) **single data source**: where one dataset is considered. (ii) **multiple data source**: where multiple data sources are considered.

2.2 Statistical Disclosure Control (SDC)

The objective of data privacy is to safeguard the collected personal data so that the data can be published or shared without exposing sensitive information or allowing the data to be linked back to the individuals who contributed in the data. Essential mechanisms to ensure privacy preservation in data publishing and sharing have been developed by SDC community. According to [Dom08], there are three key sub disciples of SDC as (i) tabular data protection, (ii) dynamic database protection, and (iii) micro–data protection.

- **Tabular–data protection**: This is crucial for statistical agencies to publish survey or census results using SDC techniques. Historically, results were shared in aggregated formats, limiting the scope for future analysis. The primary privacy requirement is to ensure that no sensitive information about individuals can be inferred from the published aggregated data.

- **Dynamic/statistical database protection**: Statistical queries are submitted by users in order to obtain aggregated information (e.g. sum, mean). The primary privacy requirement is to ensure that successive queries should not enable the inference of sensitive information about the individuals.

- **Micro–data protection**: This involves personal data collected from the individuals. The privacy requirement is to publish or share micro–data

without compromising individual's privacy. Data anonymization techniques are applied in order to protect such micro–data.

There are four types of attributes available in a micro–dataset: *(i) identifiers:* attributes that directly identify data subjects e.g., social security number or email address etc. *(ii) quasi–identifiers:* attributes, that in combination can be linked with external information to re–identify some of the respondents. For instance a combination of *(age, zipcode, gender)* can serve as a quasi–identifier in some context *(iii) confidential:* attributes containing sensitive information on the respondent such as salary or health condition *(iv) non–confidential:* attributes without including sensitive information.

2.3 Data Anonymization Techniques

Various data anonymization techniques have been introduced to protect the database. According to [Tor17], these techniques, also referred to as masking methods, can be broadly categorized into three main groups based on how they manipulate the original data to create a protected dataset.

- Perturbative: The original micro–dataset is modified or distorted to produce a protected dataset. Common techniques in this category include noise addition, microaggregation, and rank swapping.
- Non–Perturbative: The original micro–dataset values are replaced with less–specific or more general values to create a protected dataset. Common techniques in this category include generalization and suppression.
- Synthetic Data Generators: Rather than modifying the original dataset, an entirely new artificial dataset is generated to mimic the statistical properties of the original data. This synthetic dataset is then used as a substitute for the original values.

The choice of an appropriate data masking method depends on the type of database (e.g., continuous, categorical, time–series) and the database usage. Some of the most common techniques are discussed below.

2.3.1 Noise Addition and Multiplication

Noise addition and multiplication are commonly used data masking techniques, particularly for numerical data, where a certain amount of noise is added or multiplied with the original data to create a protected dataset [Bra02]. This method ensures that individual data points remain obfuscated while retaining statistical properties at a macro level. For example, if X represents the original data, the protected data X' is obtained through the following operations:

$$X' = X + \epsilon \quad \text{(Addition)} \qquad X' = X \cdot \epsilon \quad \text{(Multiplication)}$$

Here, ϵ represents the noise, which is typically drawn from a distribution with a mean of 0 and variance σ^2. If the noise is drawn from a normal distribution, it can be denoted as $\epsilon \sim N(0, \sigma^2)$. The amount of noise added is directly related to the level of privacy achieved: a higher variance in ϵ results in stronger privacy but may compromise data utility. In contrast, a lower variance preserves more of the original data but provides less privacy protection. Noise addition ensures differential privacy (which is discussed later) by masking the contribution of individual data points, making it difficult to re–identify sensitive information. Furthermore, noise multiplication introduces multiplicative perturbations, providing an alternative method of protecting data while potentially enhancing the robustness of the model to privacy attacks.

2.3.2 Swapping

Data swapping was first introduced by Dalenius [DR82] in 1978 for categorical data, with the goal of preserving the t–order frequency/contingency tables while protecting the individual data points. Over time, the concept was extended to numerical data through a technique known as rank swapping [Moo96]. In rank swapping, the values of a given variable are first sorted in ascending order, and then each ranked value is swapped with another randomly selected ranked value within a specified range p. Typically, p represents a percentage of the total number of records in X, and it allows the user to control the level of disclosure risk. The value of p is directly proportional to the level of privacy achieved: a higher p increases privacy by reducing the risk of re–identification, but may also reduce the utility of the data. It has been classified as one of the best methods for protecting micro–data in numerical attributes by [DT01a], and the best for categorical attributes by [Tor04].

2.3.3 Microaggregation

Microaggregation is another data masking technique that involves creating small micro–clusters from the original dataset. The values within each cluster are then replaced by a cluster representative, typically the mean or median of the records within the cluster. This process ensures that individual data points are obfuscated by representing them as a group. To achieve privacy, each micro–cluster must contain a predefined number of records i.e., k. As a result, the cluster representative—now the published data—is no longer a single record's value but a representation of the entire cluster. This aggregation helps mask individual data points, making it difficult to identify specific records, thereby achieving privacy while maintaining the overall statistical properties of the dataset.

This method can be formulated as an optimization problem, where the objective is to partition the data into homogeneous clusters that minimize the global error, typically measured by the sum of squared errors (SSE) between

the records and their respective cluster centers.

$$SSE = \sum_{j=1}^{D} \sum_{i=1}^{N} (x_{ij} - \bar{x_j})^2 \tag{2.1}$$

where D is the number of attributes, N is the number of records, x is the value of record, and $\bar{x}$ is the mean of $j-th$ feature. Various clustering methods have been proposed in the literature for microaggregation. Some methods use a fixed group size, where the cluster size k is predetermined and constant [DT05]. In contrast, other methods propose variable cluster sizes, where the size of each cluster is determined based on the characteristics of the data at hand [DM02].

For numerical data, the Euclidean distance is typically used to compute the distances between records, and the arithmetic mean is employed as the cluster representative. When each attribute is microaggregated independently, this approach is known as univariate microaggregation. In high–dimensional datasets, the distances between records and the cluster centroid increases, leading to significant utility loss, especially when attributes are not correlated. To address this, multivariate microaggregation could be used that groups related attributes together and independently microaggregates each group, rather than treating the entire dataset as a whole. Various heuristics have been developed for multivariate microaggregation such as MDAV (Maximum Distance to Average Vector) [Dom+06] and V–MDAV (Variable Maximum Distance to Average Vector) [SMD06]. The MDAV algorithm is described in Algorithm 1 which is also used later in Chapters 3 and 4.

2.3.4 Generalization and Recoding

In this method, a few categories are combined into more general ones to achieve data protection. This technique is primarily targeted at categorical attributes. It can be further classified into two types: global and local recoding. In global recoding, the same recoding is applied to all categories in the original data. This approach tends to result in larger information loss, as changes are applied uniformly across all categories, regardless of their specific need for privacy protection. While simple, global recoding often leads to a greater reduction in data utility. In contrast, local recoding allows different generalizations to be applied to the same category based on its occurrence in different records. This approach introduces a larger domain for the dataset, providing a finer–grained level of privacy protection. However, this can be problematic when analyzing the protected dataset, as the expanded domain might introduce complexity or inconsistencies in analysis due to varying recoding schemes applied to similar categories.

2.3.5 Suppression

The suppression method involves replacing some values with a special label. It is typically used for categorical data. It can be applied in combination

Algorithm 1 MDAV Microaggregation

Require: Original dataset $X = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^D$, Micro–cluster size k
Ensure: Anonymized dataset $X' \in \mathbb{R}^D$

while $|X| \geq 3k$ **do**
    Compute the centroid $\bar{x}$ of all records in X
    Identify the record $x_r \in X$ that is farthest from $\bar{x}$
    Identify the record $x_s \in X$ that is farthest from x_r
    Form a cluster C_r with x_r and its $k-1$ nearest neighbors
    Form a cluster C_s with x_s and its $k-1$ nearest neighbors
    Remove the records in C_r and C_s from X
end while
if $|X| \geq 2k$ **then**
    Compute the centroid $\bar{x}$ of X
    Identify the record $x_r \in X$ that is farthest from $\bar{x}$
    Form a cluster C_r with x_r and its $k-1$ nearest neighbors
    Remove the records in C_r from X
    Form a cluster C_s with the remaining records
    $C = C \cup \{C_r, C_s\}$
else
    $C \leftarrow C \cup \{X\}$
end if
return Anonymized dataset X'

with generalization to achieve the k–anonymity privacy model. In this context, suppression helps ensure that each record is indistinguishable from at least $k-1$ other records. When suppression of a particular value in a record results in the suppression of all subsequent appearances of that value across the dataset, it is referred to as global suppression. This approach guarantees that the suppressed value is uniformly hidden across all records, increasing privacy but potentially reducing data utility. On the other hand, local suppression occurs when the suppression of a value in one record is independent of its occurrence in other records. In this case, only specific instances of a value are suppressed, allowing for more flexibility in retaining data utility while still contributing to privacy protection.

2.4 Privacy Models

In the previous section, we discussed commonly used data anonymization techniques, which focus on transforming original data into a protected form to minimize disclosure risk. Complementing these techniques, privacy models provide formal definitions of privacy and outline specific conditions that, when satisfied, ensure a measurable degree of privacy while controlling disclosure risks. Anonymization techniques and privacy models are inherently interre-

lated. Anonymization techniques serve as the mechanism to fulfill the requirements of specific privacy models by defining how the original data should be transformed to produce protected data. This interplay enables data controllers to carefully balance the trade–off between privacy and utility, allowing them to fine–tune the level of privacy protection while maintaining sufficient data utility for analysis.

2.4.1 k–Anonymity

k–Anonymity is a widely adopted privacy model introduced by Samarati [SS98; Sam01] that is designed to safeguard individual privacy by ensuring that each record in a dataset cannot be distinguished from at least $k-1$ other records based on a set of quasi–identifiers. This is achieved by modifying the dataset such that any group of records sharing the same quasi–identifier values forms an equivalence class. While k–anonymity effectively prevents identity disclosure, it does not fully eliminate the risk of attribute disclosure. For instance, if all records within an equivalence class share identical values for sensitive attributes, an attacker could infer those values, leading to what is known as a homogeneity attack [De +12]. Additionally, if an attacker possesses prior knowledge about an individual and there is limited variation in the sensitive attribute values, they may exploit this information to deduce sensitive details.

To protect individual privacy, we anonymize the quasi–identifiers by applying techniques such as generalization. This ensures that the resulting database reduces the risk of identity disclosure while maintaining utility for analysis. Table 2.1 and 2.2 illustrate how generalization and suppression techniques can be applied to protect sensitive information while enforcing k–anonymity. Table 2.1 shows the original dataset, which contains attributes such as Zip Code, Birth Year, Gender, and Illness without any privacy–preserving transformations. In this format, the dataset presents a significant privacy risk, as individuals can be uniquely identified using quasi–identifier like Zip Code, Birth Year and Gender. To address this risk, Table 2.2 demonstrates $k = 2$ anonymity using generalization and suppression. Birth Year values are generalized into broader intervals, such as 1980–1985, 1985–1990 etc, ensuring that individuals within the same interval cannot be distinguished. Additionally, Zip Code values are partially suppressed by replacing the last digit with a wildcard symbol (e.g., 1234*). This reduces the risk of exact identification while still retaining some geographic context. Gender values are generalized according to the most common value in that cluster. These adjustments create equivalence classes where each group contains at least two records with identical quasi–identifiers, satisfying $k = 2$ anonymity for the list of quasi–identifiers. However, these transformations also reduce the granularity of the data, reflecting the trade–off between privacy and utility. Micro–aggregation is one of the techniques used to achieve k–anonymity by grouping similar records and replacing them with aggregated values.

Table 2.1: Original Dataset

Zip Code	Birth Year	Gender	Medical Condition
12345	1982	Male	Flu
12346	1983	Female	Cold
12447	1995	Male	Diabetes
12538	1986	Female	Asthma
12349	1984	Male	Flu
12530	1988	Female	Cold
12441	1992	Male	Diabetes

Table 2.2: Anonymized Dataset with $k = 2$

Zip Code	Birth Year	Gender	Medical Condition
1234*	1980—1985	Male	Flu
1234*	1980—1985	Male	Cold
1244*	1990—1995	Male	Diabetes
1253*	1985—1990	Female	Asthma
1234*	1980—1985	Male	Flu
1253*	1985—1990	Female	Cold
1244*	1990—1995	Male	Diabetes

2.4.2 l–diversity

k–anonymity is vulnerable to attribute inference attacks, especially when the values of sensitive attributes within an equivalence class are homogeneous. To address this issue, l–diversity [Mac+07] was proposed. In l–diversity, each group of k records must exhibit diversity in the values of sensitive attributes, ensuring that attackers cannot easily infer sensitive information based on the lack of variability. This privacy model helps to prevent attacks like the homogeneity and attribute inference attack. Several approaches have been introduced to ensure l–diversity, including distinct l–diversity, entropy l–diversity, and recursive (c, l) l–diversity. However, despite its improvements, l–diversity does not offer a comprehensive solution to eliminate all types of attribute inference attacks.

Skewness attacks can still target l–diversity. This occurs because, if certain values in a sensitive attribute are rare in the original dataset, the application of l–diversity will introduce more diverse values for those attributes. As a result, the attacker can exploit the differences in the distribution of sensitive attributes between the original and the anonymized dataset. By observing these distribution discrepancies, the attacker may link a particular individual to an equivalence class. Once the individual is associated with a specific class, there is a high probability that the attacker can identify that individual within the class, undermining the protection provided by l–diversity.

2.4.3 t–closeness

t–closeness [LLV06] is an extension of k–anonymity designed to address its vulnerability to attribute inference attacks. This model introduces a constraint on the distribution of sensitive attribute values within equivalence classes. Specifically, t–closeness requires that the distribution of sensitive values in any equivalence class is similar to their distribution in the entire dataset. While both l–diversity and t–closeness aim to minimize the risk of attribute disclosure, they have been criticized for making unrealistic assumptions about the data distribution of sensitive attributes. Additionally, applying these models often reduces data utility more significantly than k–anonymity, as they further distort the relationships between sensitive attributes and quasi–identifiers within equivalence classes.

2.4.4 Differential Privacy

The Differential privacy (DP) model, introduced by Dwork [Dwo06], has become a cornerstone in privacy–preserving data analysis. This model guarantees that the presence or absence of any single individual in a dataset cannot be inferred by analyzing the output of a function applied to two neighboring datasets. Neighboring datasets are defined as datasets that differ by only one record. DP ensures that the output of the function doesn't vary significantly, regardless of whether a specific record is included or excluded. This property provides *plausible deniability*, ensuring that the presence or absence of any particular record remains uncertain, thereby protecting individual privacy.

Definition 1. *(ϵ, δ)–Differential Privacy: Consider two datasets as neighboring if they differ by only one record (either by the addition or removal of a single data point). A mechanism F is said to be (ϵ, δ)–differentially private if, for any two neighboring datasets DB_1 and DB_2, and for any subset S of the output range of F, the following inequality holds:*

$$P[F(DB_1) \in S] \leq e^{\epsilon} \times P[F(DB_2) \in S] + \delta. \tag{2.2}$$

Here, ϵ controls the strength of the privacy guarantee and δ accounts for possibility of failure in maintaining the privacy guarantee. Typically, δ is set to small values such as $1/N$ with N representing the number of records in a dataset, with smaller values providing stronger privacy. A key property of differentially private mechanisms is that any post–processing (data–independent transformation) of their output remains differentially private with the same privacy guarantees. The above expression underlines that, e^{ϵ} is the bound of the difference between two probabilities. Thus, it becomes clear that, the smaller the ϵ, the greater the privacy. When it is equal to zero, distributions of both neighboring data sets are the same. It means there is no privacy leakage.

There are different mechanisms for implementing DP such as Laplace noise mechanism, Gaussian noise mechanism, exponential mechanism, randomized

response, sample–aggregate method, which could be used depending on the type of query and application. One of the most commonly used mechanism for numerical functions is Laplace mechanism, which is also discussed here. Laplace mechanism perturbs the data with the noise drawn from the Laplace distribution.

Definition 2. *Laplace Mechanism: Let f be a function with sensitivity Δs. Then, the function $F(x)$ defined as*

$$F(x) = f(x) + Lap(\frac{\Delta s}{\epsilon}) \tag{2.3}$$

satisfies ϵ–differential privacy.

Here, ϵ is the privacy budget, Δs is the global sensitivity of the function f, and $Lap(\mathrm{S})$ denotes sampling from Laplace distribution with center 0 and scale S. The scale S of noise is calibrated to the sensitivity of f for two neighboring databases and the privacy requirements. Global sensitivity is the maximum variation a given function takes with respect to all neighboring datasets. Mathematically, it is defined as

$$\Delta s = \max_{DB_1, DB_2} \|F(DB_1) - F(DB_2)\|_1 \tag{2.4}$$

where $\|\cdot\|_1$ represents the L_1 norm, which is the sum of the absolute differences between corresponding elements, and DB_1 and DB_2 are arbitrary neighboring datasets. In machine learning, DP can be introduced at various stages, such as at the input level, during model training (DP–Training), or when serving predictions (model inference) [Pon+23]. For non–convex loss functions, one of the most effective DP–Training methods is gradient–noise injection, such as Differentially Private Stochastic Gradient Descent (DPSGD) [Aba+16]. This approach limits the sensitivity of the loss function by clipping per–example gradients and adding Gaussian noise to the aggregated clipped gradients to ensure privacy. The noise level is tied to the clipping norm (which controls sensitivity) and the strength of the ϵ guarantee. A similar strategy could be adopted to other optimizers as well.

2.5 Synthetic Data Generation

Masking methods, both perturbative and non–perturbative, modify the original data to ensure confidentiality. A promising alternative to these approaches is the release of synthetic data. In this approach, a model is trained on the original data, and synthetic values are generated by sampling from this model. Depending on the desired level of privacy protection, either a subset of records (partially synthetic data) or the entire dataset (fully synthetic data) is replaced with synthetic values. According to [Tor22], the methods for generating synthetic data can be categorized into three major classes based on their underlying principles and techniques.

1. **Synthetic reconstruction** methods leverage a dataset containing the marginal distribution of the entire population and the conditional probabilities for specific attributes, often derived from publicly available contingency tables. The data generation process follows a structured approach. (i) Individuals are either selected from an existing population or synthetically created to represent the target demographic. (ii) Attributes are sequentially assigned to the synthetic records. Each attribute is generated one at a time, ensuring consistency with the relevant conditional probabilities. For example, after creating a synthetic individual, an attribute such as residential status (e.g., homeowner or tenant) is assigned. Based on this value, subsequent attributes, like property size, are generated in alignment with the conditional probabilities. (iii) Historically, methods like Iterative Proportional Fitting (IPF), developed in the 1930s, were employed to construct such datasets by aligning marginal and conditional distributions [BT13; HW01]. More advanced techniques have since been developed to improve the efficiency and accuracy of the synthetic data generation methods.

2. **Combinatorial optimization** methods generate synthetic data by systematically combining attribute values from the original dataset. Instead of replicating statistical patterns directly, this method focuses on creating all possible combinations of attributes or a representative subset, ensuring diversity and offering robust privacy guarantees. It is particularly effective for datasets with categorical variables or when ensuring diversity in the synthetic data is crucial. By generating combinations that may not exist in the original dataset, it inherently enhances privacy by reducing the risk of re–identifying individuals in the original data.

3. **Model–based simulations** have gained significant momentum in synthetic data research, driven by advancements in generative models like Generative Adversarial Network (GANs). These models generate synthetic data by learning the underlying distribution of real datasets. Over time, various GAN variants have been developed to cater to specific data types and applications. CTGAN (Conditional Tabular GAN) is specialized for generating high–quality tabular data with mixed data types and imbalanced distributions. Another GAN, i.e., tableGAN is designed to produce realistic tabular data by learning row–wise dependencies. DPGAN (Differentially Private GAN) integrates differential privacy mechanisms to ensure the generated data maintains privacy guarantees. PATEGAN (Private Aggregation of Teacher Ensembles GAN) focuses on privacy–preserving synthetic data for tabular datasets using a teacher–student framework. In addition to GANs, other model–based generators have proven effective such as Variational Auto–Encoders (VAE) and Diffusion Models. We describe some of them as follows.

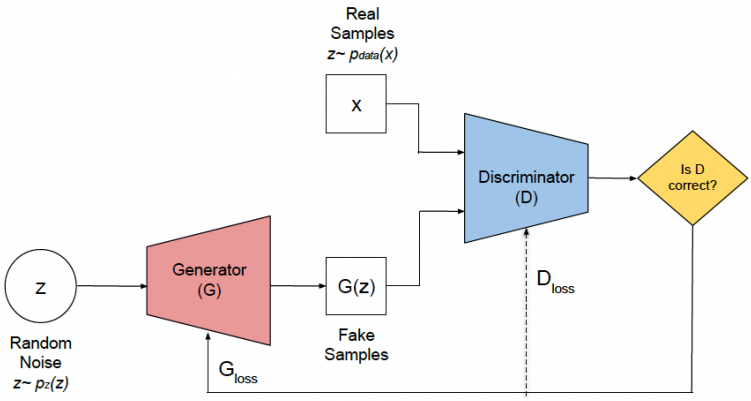


Figure 2.1: Generative Adversarial Network Architecture

2.5.1 Generative Adversarial Network (GANs)

GANs are a class of deep learning models designed to generate realistic synthetic data by learning the underlying data distribution. Introduced by [Goo+20] in 2014, GANs consist of two neural networks: the *generator* and the *discriminator*. The *generator* takes random noise (sampled from a latent space) as input and produces synthetic data. Its goal is to generate data that closely resembles the real data. The *discriminator* is a binary classifier that differentiates between real data and data generated by the generator. Its goal is to correctly identify whether the input is real or fake. The training process involves the two networks playing a *min-max game*, where the generator tries to fool the discriminator, and the discriminator tries to distinguish the real data from the fake data. The aim of *generator* is to minimize the *discriminator's* ability to correctly identify fake data. The generator's loss is defined as:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z(z)} \left[\log D(G(z)) \right] \tag{2.5}$$

Here, z is the input noise, $G(z)$ is the generated data, and $D(G(z))$ is the discriminator's probability of classifying the generated data as real. The aim of *discriminator* is to maximize its ability to correctly classify both real and fake data. The discriminator's loss is defined as:

$$\mathcal{L}_D = -\mathbb{E}_{x \sim p_{data}(x)} \left[\log D(x) \right] - \mathbb{E}_{z \sim p_z(z)} \left[\log(1 - D(G(z))) \right] \tag{2.6}$$

Here, x is a real data sample, and $D(x)$ is the discriminator's probability of classifying x as real. During training, the *generator* improves its outputs based on feedback from the *discriminator*, while the *discriminator* adapts to the *generator's* improvements. Ideally, the *generator's* outputs become indistinguishable

from real data, and the *discriminator's* accuracy converges to 50%, indicating it can no longer distinguish the real data from the fake data. The architecture of GANs is pictorially depicted in Figure 2.1. GANs have been successfully applied to various domains, including image synthesis, text–to–image generation, and synthetic data generation, establishing them as a cornerstone of modern generative modeling. We will use different architectures of GANs in Chapter 4.

2.5.2 Variational Autoencoder (VAE)

Variational Autoencoder (VAE) are a type of generative model that learn to encode the data into a latent representation and then decode it back to reconstruct the original data. Introduced by [Kin13] in 2013, VAE combine probabilistic modeling with neural networks, making them effective for generating new samples from a learned distribution. It consists of two components: the *encoder*, and the *decoder*. The *encoder* maps the input data x into a latent space by learning the parameters of a probability distribution. The latent representation z is sampled from this distribution:

$$q_\phi(z|x) \sim \mathcal{N}(\mu(x), \sigma^2(x)) \tag{2.7}$$

where $\mu(x)$ and $\sigma^2(x)$ are the mean and variance learned by the encoder network with parameters ϕ. The *decoder* maps the latent variable z back to the data space to reconstruct the input. It learns the conditional probability $p_\theta(x|z)$, parameterized by θ. The VAE optimizes a loss function that is a combination of reconstruction loss and KL divergence loss. The reconstruction loss measures how well the reconstructed data $\hat{x}$ matches the input x, typically using the negative log–likelihood:

$$\mathcal{L}_{\text{reconstruction}} = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \tag{2.8}$$

While KL Divergence loss regularizes the latent space by encouraging the approximate posterior $q_\phi(z|x)$ to be close to the prior distribution $p(z)$, typically a standard normal distribution:

$$\mathcal{L}_{\text{KL}} = D_{KL}(q_\phi(z|x) \| p(z)) \tag{2.9}$$

The total loss is the sum of these two terms:

$$\mathcal{L} = \mathcal{L}_{\text{reconstruction}} + \mathcal{L}_{\text{KL}} \tag{2.10}$$

Figure 2.2 depicts the architecture of VAE. During training, the *encoder* and *decoder* are optimized jointly to minimize the total loss. VAE excel at generating smooth interpolations in the latent space and are widely used in applications such as image synthesis, anomaly detection, and representation learning.

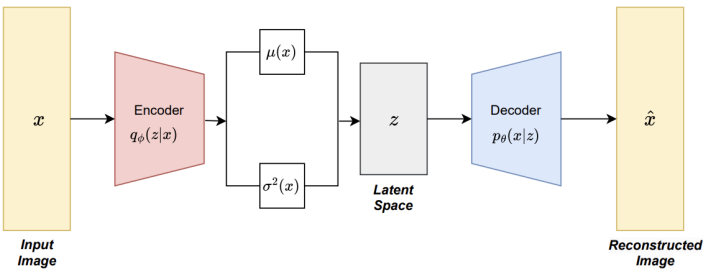


Figure 2.2: Variational Autoencoder Architecture

2.6 Selection of a Data Protection Mechanism

Different data protection methods have distinct properties and introduce varying degrees of distortion to the original data. The extent of this distortion depends on both the chosen method and its parameterization. A critical consideration when selecting a protection mechanism is the trade–off between privacy and data utility—specifically, the level of protection provided and the extent to which the modified data remains useful for analysis. This trade–off can be assessed using two key parameters:

1. Disclosure Risk: It measures the probability that sensitive information can still be inferred despite the applied protection mechanism.

2. Information Loss: It computes the degree to which data utility is degraded due to the distortion introduced by the protection technique.

An optimal data protection method should minimize the disclosure risk while preserving as much information as possible, ensuring that the protected data remains suitable for meaningful analysis. We discuss both of them in detail.

2.6.1 Disclosure Risk

Disclosure occurs when an adversary leverages observations and analysis of a released dataset to enhance their knowledge about a specific item of interest. A privacy violation arises when an adversary exploits either the raw personal data or the outcomes of an analysis performed on that data to infer previously unknown confidential attributes. This threat is known as disclosure risk. There are two main types of disclosure risk.

1. **Identity Disclosure** occurs when an attacker uses publicly available database or introduces own database to link the records and identify an

individual in the protected database. This is also known as re–identification, and could be achieved through record linkage between the protected and the original database.

2. **Attribute Disclosure** occurs when an attacker infers sensitive attributes of the target individual, even if their identity remains undisclosed. This type of disclosure arises when an adversary, possessing prior domain knowledge about a target individual, leverages protected data to gain additional insights into their sensitive attributes.

2.6.2 Information Loss

A data protection method alters the properties of the original database, impacting its quality and utility. This modification degrades the accuracy of any analysis performed on the protected data, a phenomenon known as information loss. High information loss reduces the dataset's analytical utility, limiting its effectiveness for meaningful insights. Conversely, low information loss may result in an increased disclosure risk, as the protected data retains more identifiable patterns from the original dataset. Therefore, an optimal trade–off between privacy and utility must be established to ensure both effective data protection and usability for analysis.

Information loss can be categorized as generic or specific. Generic information loss is assessed using statistical properties of the data, such as individual record values [DT01b; DT01a], value rankings [LWZ08], or summary statistics like mean, variance, and covariance [MS05]. In this thesis, we focus on specific information loss, particularly in the context of machine learning tasks. To quantify this, we evaluate the performance of machine learning models, such as classification, regression, or clustering on both the original and protected datasets. The information loss is then measured using performance metrics such as classification accuracy or regression errors, providing a direct assessment of how data protection affects the model effectiveness.

2.7 Machine Learning for High–Dimensionality

High–dimensionality presents challenges for both the data and the models. Real–world datasets often contain a large number of attributes, resulting in high–dimensional feature spaces. Simultaneously, machine learning models designed to process such data are often large–scale, with millions of parameters. As a result, high–dimensionality impacts both the data representation and the model complexity. In this section, we discuss various techniques that address such concerns, and we use them in later chapters.

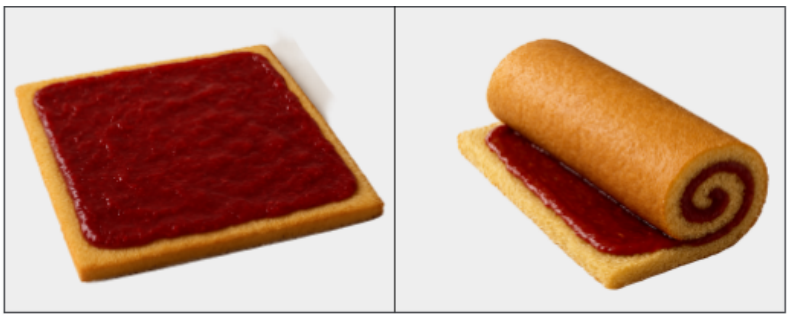


Figure 2.3: Manifold Hypothesis

2.7.1 Manifold Learning

A manifold is defined as a topological space that locally resembles Euclidean space. In small neighborhoods, it can be approximated by the Euclidean space. More precisely, a manifold is a space in which every point has a neighborhood that is homeomorphic to an open subset of an n–dimensional Euclidean space. This concept can be formally described using the following definition.

Manifold Learning: Given a finite set of data points $x_1, ...x_n \in \mathbb{R}^D$ *in a* D*–dimensional space, a Manifold learning algorithm aims to find the points* $y_1, ...y_n \in \mathbb{R}^d$ *in low–dimensions where* $d \ll D$ *such that Euclidean relationship between* (y_i, y_j) *reflects the intrinsic non–linear relationship between* (x_i, x_j) *[TSL00].*

Figure 2.3 illustrates the swiss roll cake with a jam topping. The jam layer–the most flavorful and important part of the cake, lies on a flat 2D surface before the cake is rolled. Once rolled, this layer becomes embedded in a 3D spiral structure. This captures the essence of the manifold hypothesis, which states that data points lie on a low–dimensional manifold denoted by $\mathbb{R}^d$, which is embedded in a high–dimensional space denoted by $\mathbb{R}^D$. The goal of manifold learning is to find a way to map the data from a high–dimensional space, $\mathbb{R}^D$, to a lower–dimensional space while keeping the geometric properties as much as possible. As dimensions increase, a larger proportion of the data tends to reside near the corners of the feature space [Spr14], complicating effective analysis. High–dimensional data is present in various forms, for instance tabular datasets with numerous rows and columns, image data, and textual data. For low–dimensional data, visualization through graphical plots effectively reveals local geometric patterns. However, these techniques are not as intuitive or feasible for high–dimensional data. To address this limitation, it is essential to understand the structure and the geometry of high–dimensional data and transform high–dimensional data into lower–dimensional representations, with the help of manifold learning. One common method is to preserve the pairwise distances between data points during this transformation. This involves calculating the distances between points in the original high–dimensional space

and ensuring these distances are maintained in the lower–dimensional space. By doing this, manifold learning captures the key geometric patterns in the data, making it easier to interpret and remain useful for tasks like clustering, classification, and visualization.

2.7.2 Euclidean vs Geodesic Distance

When manifold exhibits a curvature–like structure, small distances along the curve in a high–dimensional space can appear as much larger distances on the manifold itself. Mathematically, the Euclidean distance is defined as follows.

Definition 3. *Euclidean Distance: Let $\mathbb{R}^D$ denote an D–dimensional Euclidean space. Consider two points $x = (x_1, x_2....x_D)$ and $y = (y_1, y_2....y_D)$ in $\mathbb{R}^D$. The Euclidean distance between x and y is defined as.*

$$dist_{Euclidean}(x, y) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2} \tag{2.11}$$

This distance measures the length of the shortest path between the points x and y in the Euclidean space $\mathbb{R}^D$.

In a manifold, Euclidean distance fails to capture the true geometry of the data points because it only considers linear paths and does not account for the manifold's curvature. In contrast, geodesic distance provides a more accurate measure by considering the shortest path along the curved surface, following the actual structure of the manifold. This is done by finding the path between the points that minimizes the distance, rather than relying on straight lines, which better reflects the true relationships between points on a manifold. Mathematically, geodesic distance can be defined as follows.

Definition 4. *Geodesic Distance: Let $\mathbb{M}$ denote a d–dimensional manifold. Consider two points m_1, $m_2 \in \mathbb{M}$ and a smooth path γ: $[0, 1] \to \mathbb{M}$ such that $\gamma(t) \in \mathbb{M}$, $\gamma(0) = m_1$ and $\gamma(1) = m_2$. The derivative $\gamma'(t)$ depicts the velocity of gamma since it passes through the point $\gamma(t)$, with also $\gamma'(t) \in \mathbb{M}$. The length of the curve $L(\gamma)$ is defined as:*

$$L(\gamma) = \int_0^1 \langle \gamma'(t), \gamma(t) \rangle^{1/2} dt \tag{2.12}$$

where $\langle \cdot, \cdot \rangle$ represents the inner product between two vectors. The distance between points m_1 and m_2, i.e., $\rho(m_1, m_2)$ is infimum over all possible paths connecting the two points m_1 and m_2. If this distance is achieved by a particular path γ, we say that γ is a geodesic [Lee18; RBS21].

The Euclidean and Geodesic distances can be computed, as shown in Figure 2.4. Imagine data samples represented by points in a high–dimensional

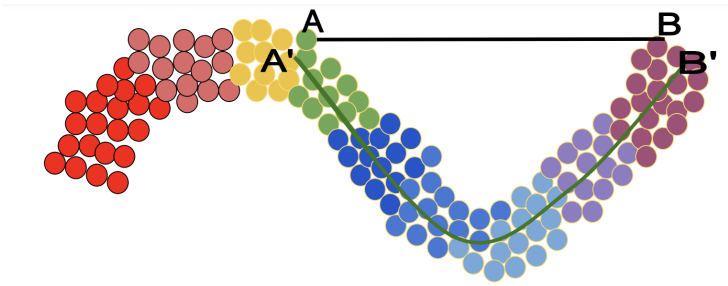


Figure 2.4: Euclidean vs Geodesic Distance

space. The task is to compute both the Euclidean and Geodesic distances between two points, A and B. The Euclidean distance is simply the straight–line distance between A and B, representing the shortest possible path between the two points. However, this approach does not account for the actual structure or geometry of the data points and can ignore local variations in the data, potentially leading to the loss of important information. This issue is especially problematic in high–dimensional spaces, where data points may be sparse and not uniformly distributed, making the Euclidean distance less informative.

To preserve the local geometry of the data and find the true shortest path between points, Geodesic distance is a better choice. Unlike Euclidean distance, Geodesic distance between points A' and B' takes into account the structure of the data by calculating the shortest path along the manifold formed by adjacent data points. This path tries to ensure that the relationships between points are well represented. In later chapters, we build on this concept by using geodesic distance to measure the distance between data points in high–dimensional space. This approach assumes that a sufficient number of intermediate points exist to approximate the true geodesic path; otherwise, the geodesic and Euclidean distances would be nearly identical.

2.7.3 Manifold Learning Techniques

Manifold learning techniques are generally divided into two broad categories: linear and non–linear techniques. Linear manifold learning methods assume that the high–dimensional data lies on a linear subspace, meaning the relationships between data points can be adequately described by linear mappings. These techniques perform well on the datasets that exhibit an inherent linear structure and can successfully compute a low–dimensional embeddings. Popular linear techniques for dimensionality reduction include Principal Component Analysis (PCA) [Hot33], Multi–Dimensional Scaling (MDS) [Kru64], and Linear Discriminant Analysis (LDA) [Fis36]. These methods focus on preserving the linear relationships between data points while reducing dimensionality. However, when the data lies on a non–linear manifold, these linear

Algorithm 2 Isometric Mapping (ISOMAP)

Require: Data points $X = \{x_1, x_2, \ldots, x_N\}$ in $\mathbb{R}^D$, neighborhood size k or threshold ϵ, target dimension d.

Ensure: d–dimensional embedding $Y = \{y_1, y_2, \ldots, y_N\}$ in $\mathbb{R}^d$.

1: Construct Neighborhood Graph G by connecting points i and j (measured by $Dist_X(i,j)$) if they are: within ϵ distance (ϵ–ISOMAP), or among k nearest neighbors (k–ISOMAP).

2: Compute Shortest Paths by initializing $Dist_G(i,j) = Dist_X(i,j)$ if i and j are connected, else $Dist_G(i,j) = \infty$.

3: Update shortest paths using Floyd–Warshall:

$$Dist_G(i,j) = \min\{Dist_G(i,j), Dist_G(i,k) + Dist_G(k,j)\} \quad \forall k = 1, 2..N.$$

4: Let λ_p be the p–th eigenvalue and v_p^i be i–th component of the p–th eigenvector of matrix $\tau(Dist_G) = -HDist_{ij}^2 H/2$ where H is the centering matrix $H = I_N - 1/N e_N e_N^T$ with I as an identity matrix and $e_N = [1...1]^T \in \mathbb{R}^N$.

5: Construct d–dimensional Embedding by setting p–th component of d–dimensional vector $y_i = \sqrt{\lambda_p} v_p^i$.

techniques fail to capture the intrinsic non–linear structure, leading to distortions in the data representation and a loss of important relationships, such as the preservation of pairwise distances between points [Jol02]. Non–linear manifold learning methods are specifically designed to address this limitation. These techniques are capable of capturing and preserving the non–linear structure of the data in high–dimensional space, ensuring that the data's geometric properties, such as local neighborhoods and pairwise distances, are better represented in lower–dimensional embeddings. There are some widely used techniques including Isometric Mapping (ISOMAP), Locally Linear Embedding (LLE), t–Stochastic Neighbor Embedding (t–SNE) and Uniform Manifold Approximation and Projection (UMAP) which we use later in Chapters 3 and 4. These techniques help in generating low–dimensional embeddings of the data while preserving the manifold structure and are discussed as follows.

ISOMAP

Isometric Mapping is a non–linear dimensionality reduction technique that projects high–dimensional data onto a lower–dimensional space while preserving its intrinsic geometry [TSL00]. Unlike linear methods that rely on Euclidean distance, ISOMAP uses geodesic distance to capture the manifold's curvature. It constructs an adjacency graph that connects neighboring points and computes geodesic distances using shortest–path algorithms like Dijkstra's [Dij22] or Floyd–Warshall [Flo62]. By preserving the global structure of the manifold, ISOMAP uncovers non–linear degrees of freedom, achieves globally optimal solutions, and asymptotically converges to the true manifold as data size in-

Algorithm 3 Locally Linear Embedding (LLE)

Require: Data points $X = \{x_1, x_2, \ldots, x_N\}$ in $\mathbb{R}^D$
Ensure: Low–dimensional embedding $Y = \{y_1, y_2, \ldots, y_N\}$ in $\mathbb{R}^d$.
1: Find the nearest neighbors of each data point x_i.
2: Compute weights w_{ij} by minimizing the reconstruction error:

$$\min_w \sum_{i=1}^{N} \left\| x_i - \sum_{j=1}^{N} w_{ij} x_j \right\|^2$$

Every point x_{ij} is a linear combination of its neighbours and weights w_{ij} are computed such that x_i is close to $\sum_{j=1}^{k} w_{ij} x_j$.
3: Map the data points onto y by preserving the weights:

$$\min_y \sum_{i=1}^{N} \left\| y_i - \sum_{j=1}^{N} w_{ij} y_j \right\|^2$$

4: Return the low–dimensional embedding Y.

creases, making it effective for complex datasets. The ISOMAP algorithm is described in Algorithm 2. We will use this algorithm later in Chapter 3.

Locally Linear Embedding

Locally Linear Embedding (LLE) [RS00] is a non–linear dimensionality reduction technique that focuses on preserving the local geometry of data by assuming that each data point and its neighbors lie on a locally linear manifold. The method works by reconstructing each data point as a linear combination of its nearest neighbors, typically determined using Euclidean distance. After this, LLE projects the data points into a lower–dimensional space while maintaining the relationships between the neighbors. This approach belongs to the broader category of local linear transformations and is particularly effective for datasets that exhibit smooth, open planar manifolds, where the underlying structure can be approximated well by linear combinations in the local neighborhood. The LLE algorithm is described in Algorithm 3. We will use this algorithm later in Chapter 3.

t–SNE

The t–SNE (t–Distributed Stochastic Neighbor Embedding) [VH08] algorithm measures pairwise similarities between data points in the high–dimensional space by representing them as probabilities that reflect how likely it is for one point to be a neighbor of another. It then constructs a similar probability dis-

tribution for the points in the lower–dimensional space. To ensure that the two distributions align, t–SNE minimizes their divergence, typically measured via the Kullback–Leibler (KL) divergence using gradient descent. This optimization process enables t–SNE to effectively preserve the local structure of the data while revealing patterns and clusters in the reduced–dimensional embedding.

UMAP

UMAP (Uniform Manifold Approximation and Projection) [MHM18] models the data as a weighted graph where edges represent the local structure of the data. It builds a high–dimensional graph based on local neighborhoods using a fuzzy simplicial set, then optimizes a lower–dimensional graph to preserve the topological structure. This optimization minimizes the cross–entropy between the two graphs, preserving both local and some global structures. UMAP is able to accelerate the optimization and preserve much more global structure than t–SNE.

2.7.4 Model Compression for Language Models

Large Language Models (LLMs) have transformed the field of Natural Language Processing (NLP), excelling in tasks like question answering, language translation, content generation, and sentiment analysis, enabling AI to interact with humans in natural language effectively. These models, with billions or even trillions of parameters, demonstrate superior performance. According to the scaling laws: the larger the model, the better it performs. However, deploying such massive models for real–world applications is challenging due to their high computational demands, requiring substantial memory, multiple GPUs, and considerable energy resources, which also raise environmental concerns. Model compression techniques address these limitations by reducing the model size and computational requirements of LLMs. By enabling faster inferences and lowering costs, model compression brings the power of LLMs to everyday applications, bridging the gap between cutting–edge AI and practical usability. Common model compression techniques include Knowledge Distillation, Pruning and Quantization which are discussed as follows. We later use them in Chapter 5.

Knowledge Distillation

Knowledge distillation [HVD15] is a technique for transferring the knowledge from a large, complex model (the teacher) to a smaller, more light–weight model (the student). This approach is especially beneficial when the teacher model, with its high number of parameters and computational requirements, is unsuitable for deployment in resource–limited environments. The student model is trained to replicate the behavior of the teacher by learning from its outputs, often represented as softened probability distributions over classes. Let $p_i^{teacher}$ denote the probability assigned by the teacher model to class i and

$p_i^{student}$ denote the corresponding probability assigned by the student model. These probabilities are obtained through the softmax function applied to the logits produced by each model. Mathematically,

$$p_i^{\text{teacher}} = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \qquad p_i^{\text{student}} = \frac{\exp(\tilde{z}_i/T)}{\sum_j \exp(\tilde{z}_j/T)}$$

where z_i and $\tilde{z}_i$ represent the logits for class i generated by the teacher and student models respectively, and T is the temperature parameter controlling the softness of the distributions. The distillation loss aims to minimize the discrepancy between the teacher and student probabilities. A common formulation for the distillation loss involves the KL divergence between the teacher and student distributions

$$\mathcal{L}_{\text{distill}} = \text{KL}(p^{\text{teacher}} || p^{\text{student}}) = \sum_i p_i^{\text{teacher}} \log \frac{p_i^{\text{teacher}}}{p_i^{\text{student}}} \tag{2.13}$$

The distillation loss guides the student model to align with the soft targets produced by the teacher model, capturing the fine–grained decision boundaries and complex relationships inherent in the teacher's predictions. By combining this distillation loss with the task–specific loss function (e.g., cross–entropy loss), the student model effectively absorbs the teacher's knowledge. This dual optimization enables the student to achieve a balance between compactness and performance, making it a resource–efficient yet capable alternative for deployment in environments with limited computational resources.

Pruning

Pruning [ZG17] is a technique used to enhance model efficiency by identifying and removing unnecessary or redundant parameters that have minimal impact on the model performance. The primary objective of pruning is to optimize the models for memory efficiency, faster inference, and lower energy consumption while maintaining comparable performance levels. It directly eliminates less important connections or structures within the network, offering a more interpretable and flexible approach to model compression. This method strikes an effective balance between model accuracy and model size, making it a preferred choice for resource–constrained deployments.

Various pruning strategies exist, each tailored to different stages of the model life cycle. Unstructured pruning [Kwo+20] removes individual weights based on their magnitude, creating sparse matrices that can be challenging to optimize on standard hardware. In contrast, structured pruning [WWL19] eliminates entire components, such as neurons, filters, or attention heads, resulting in a smaller, denser network that is more hardware–friendly and easier to accelerate. Pruning can be applied at different stages: pre–training pruning (before training), gradual pruning (during training), or post–training pruning (after training). The choice of pruning strategy and timing often depends on

the desired balance between performance and resource efficiency for a specific application. Common techniques include magnitude–based pruning [Lee+20], which removes parameters with the smallest absolute values, assuming they have minimal impact on the output, and gradient–based pruning [Yeo+21], which leverages gradient information to identify and eliminate the least important parameters. Pruning often involves iterative cycles of pruning and retraining to recover any lost performance, ensuring the pruned model remains effective while being significantly more efficient.

Quantization

Quantization [Wu+16; Gon+14] is a method that reduces the precision of weights and activations in a model. The objective is to reduce memory footprint and improve inference speed, which could be achieved by representing numbers with fewer bits. For instance, instead of representing weights and activations using 32–bit floating–point numbers, quantization can help in representing them using 8–bit integers. While this reduction in precision can result in some loss of model utility, methods like post–training quantization and quantization–aware training are designed to mitigate this trade–off, ensuring that the model maintains a balance between efficiency and performance.

Chapter 3

Privacy–Preserving Manifold Learning

> Anyone who steps back for a minute observe our modern digital world might conclude that we have destroyed our privacy in exchange for convenience and false security
>
> — *John Twelve Hawks*

In this chapter, we explore and propose novel methods to protect the sensitive information embedded within high–dimensional spaces. We study our first Research Question (**RQ1**) in this chapter and explore how effective are existing privacy models such as k–anonymity, differential privacy and their combinations in preserving the privacy and utility of high–dimensional datasets. To do this, we first introduce the M–MDAV privacy mechanism, which is built upon the k–anonymity framework to anonymize high–dimensional datasets effectively. To enhance this approach, we propose integrating k–anonymity with manifold learning techniques, recognizing the importance of preserving the intrinsic structure of the data while ensuring privacy. However, selecting the most suitable privacy model can be challenging, as the effectiveness of these models is highly dependent on the specific characteristics of the data and the intended use cases. To address this challenge, we introduce a hybrid privacy model, the (β, k, ϵ_0)–anonymization technique, which combines the strengths of both k–anonymity and differential privacy. This hybrid method ensures that the benefits of both approaches are preserved while satisfying their respective privacy guarantees, thereby achieving a balanced trade–off between privacy preservation and utility.

3.1 K–Anonymous Manifold Learning

We evaluate the effectiveness of the k-anonymity privacy model in preserving privacy for high–dimensional data. To achieve k–anonymity, we applied microaggregation and implemented the MDAV (Maximum Distance to Average Vector) mechanism as described in Algorithm 1 in Chapter 2. The MDAV mechanism uses distance–based aggregation, grouping records that are similar to each other. This approach preserves some of the statistical properties of the data better than other methods such as random partitioning, generalization or suppression. Additionally, the MDAV mechanism supports multi–variate microaggregation making it well suited for high–dimensional data. To further enhance its effectiveness, we propose an improved version of the MDAV mechanism to achieve k–anonymity for high–dimensional records.

We introduce the first approach, M–MDAV (Manifold–Maximum Distance to Average Vector) algorithm, a manifold–based adaptation of the traditional MDAV heuristic technique for achieving k–anonymity which is presented in Algorithm 4. Traditional distance metrics, such as Euclidean distance, Manhattan distance, or Mahalanobis distance, are commonly used to compute the distances between data points. As we have discussed in Chapter 2, in high–dimensional spaces, these metrics often lose their effectiveness in accurately measuring similarity, as they fail to account for the underlying data structure. To address this limitation, the M–MDAV algorithm employs geodesic distance (see Definition 4). Geodesic distance considers the manifold structure of the data by accounting for local neighborhoods and computing the actual shortest paths between points along the data manifold. This approach enables the algorithm to preserve the intrinsic geometric structure of high–dimensional data, resulting in a more meaningful anonymization while maintaining the data's manifold characteristics. In Algorithm 4 M–MDAV operates as follows: initially pairwise–distances between each data points are computed using geodesic distance. Then, the median of all data points is obtained by minimizing the geodesic distance between the data points. After that, clusters are formed around the data points that are furthest from the median by calculating the geodesic distance. This process is repeated until all the points get clustered. Finally, the clustered data points are replaced by the median of that cluster. This method ensures that the dataset is anonymized, preserving both privacy and the manifold structure of the data. The protected dataset can then be used for further analysis, ensuring that privacy is maintained without compromising the utility of the data.

As we will see in our experiments our algorithm M–MDAV, is not efficient enough to anonymize the high–dimensional data without loosing a lot of utility. Thus, we propose another two methodologies M–ISOMDAV and M–LLEMDAV, that uses ISOMAP and LLE as manifold learning techniques (as discussed in Chapter 2) for preserving the manifold structure of the data. These techniques helps to transform the high–dimensional data to a lower–dimensional space, where the data's inherent geometric structure is more easily preserved.

Algorithm 4 M–MDAV

Require: Original dataset $X = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^D$, Micro–cluster size k
Ensure: Protected dataset $X' \in \mathbb{R}^D$

while $|X| \neq 0$ **do**
  if $|X| \geq 3k$ **then**
    Identify median of all the records x_{median} such that :

$$x_{median} = \arg\min_{x \in X} \sum_{i=1}^{N} \gamma(x, x_i)^2 \quad \text{where } \gamma\text{: geodesic distance}$$

    Let $x_r \in X$ be the record farthest from x_{median}
    Let $x_s \in X$ be the record farthest from x_r
    Form cluster C_r with x_r and its $k-1$ nearest neighbors
    Form cluster C_s with x_s and its $k-1$ nearest neighbors
    Update dataset: $X \leftarrow X \setminus \{C_r \cup C_s\}$
    Update clusters: $C \leftarrow C \cup \{C_r, C_s\}$
  else if $|X| \geq 2k$ **then**
    Find x_{median} with all the records in X.
    Find most distant record x_r from x_{median}.
    Form cluster C_r with x_r and its $k-1$ nearest neighbors
    Form cluster C_s with the remaining records in $X \setminus C_r$
    Update clusters: $C \leftarrow C \cup \{C_r, C_s\}$
  else
    Add remaining records as a final cluster: $C \leftarrow C \cup \{X\}$
  end if
end while
Produce k–anonymized matrix X' from clusters C.

Once the data is transformed, M–MDAV algorithm is applied to anonymize the low–dimensional data points. This two–step approach effectively integrates manifold learning for preserving data structure with M–MDAV for anonymization, striking a balance between minimizing utility loss and ensuring privacy. We explored how manifold learning techniques could complement privacy mechanisms, an area that, to our knowledge, has not been extensively investigated in the literature. While other dimensionality reduction techniques, such as PCA, have been explored, they do not work well for non-linear data structures. In contrast, manifold learning is better suited for handling such structures.

In Algorithm 5, M–ISOMAP, we first take a high–dimensional dataset, apply the ISOMAP manifold learning technique to compute a lower-dimensional representation of the dataset while preserving the geodesic distances between data points. Unlike linear methods like PCA, ISOMAP takes into account the global manifold structure of the data. It works by constructing a neighborhood graph of the data points, where each edge represents a pairwise distance be-

Algorithm 5 M–ISOMDAV

Require: Data points $X = \{x_1, x_2, \ldots, x_N\}$ in $\mathbb{R}^D$

Ensure: Anonymized lower-dimensional representation $Y' = \{y_1, y_2, \ldots, y_N\}$ in $\mathbb{R}^d$ where $d < D$

1: Create a weighted graph M by connecting points x_i and x_j if their Euclidean distance $dist_E(x_i, x_j) \leq \epsilon$. Set the edge weights as $dist_E(x_i, x_j)$.
2: Compute the pairwise geodesic distance matrix $M' \in \mathbb{R}^{N \times N}$ using Dijkstra's shortest path algorithm on graph M.
3: Construct the centering matrix $H = I_N - 1/N e_N e_N^T$ and $e_N = [1....1]^T \in \mathbb{R}$.
4: Compute Kernel Matrix: $K = -\frac{1}{2} H M'^2 H$, where M'^2 is the element-wise squared matrix.
5: Perform eigen decomposition on K and select the top d eigenvalues $\{\lambda_1, \ldots, \lambda_d\}$ and their corresponding eigenvectors $\{\nu_1, \ldots, \nu_d\}$.
6: Form the lower-dimensional embedding $Y \in \mathbb{R}^{N \times d}$ such that each row $y_i = [\sqrt{\lambda_1}\nu_1^i, \ldots, \sqrt{\lambda_d}\nu_d^i]$.
7: Apply M–MDAV to Y to achieve k–anonymity and obtain the final anonymized dataset Y'.

tween neighboring points. ISOMAP then computes the shortest paths between points on this graph, which helps to preserve the manifold's intrinsic geometry. By projecting the data onto a lower-dimensional space that maintains these geodesic distances, ISOMAP effectively captures the underlying structure of non–linear high–dimensional data. After applying ISOMAP, the resulting lower–dimensional data points are then anonymized using the M–MDAV algorithm, ensuring both the privacy and the preservation of the data's manifold structure. This combined approach results in a more effective anonymization process, with minimal loss of utility.

Similarly, in Algorithm 6, M–LLEMDAV, we utilize Locally Linear Embedding (LLE) as a manifold learning technique before applying the M–MDAV algorithm for anonymization. It aims to preserve the local geometric structure of high–dimensional data by reconstructing each data point as a weighted linear combination of its nearest neighbors. The process begins by identifying the nearest neighbors for each data point based on geodesic distance. Then, reconstruction weights are computed by minimizing the error in approximating each data point using its neighbors while ensuring invariance to transformations. Finally, LLE maps the data to a lower–dimensional space by preserving these reconstruction weights, effectively maintaining local relationships. By applying LLE before anonymization, M–LLEMDAV ensures that the intrinsic structure of the data is retained while reducing dimensionality. The transformed data is then anonymized using M–MDAV, combining the advantages of structure-preserving manifold learning with privacy protection to minimize utility loss. We chose LLE for manifold learning because it effectively preserves the local geometric structures of high–dimensional data. This property is crucial for

anonymization as preserving local structure ensures that similar records remain close to each, thereby reducing distortions in data utility.

3.1.1 Experimentation and Results

We conducted experiments on the three proposed methodologies: M–MDAV, M–ISOMDAV, and M–LLEMDAV, which aim to anonymize high–dimensional data using the k–anonymity privacy model. For empirical evaluation, we utilized three different types of datasets: tabular, image, and textual datasets. We treated all attributes as quasi–identifiers, so we anonymized the entire dataset accordingly. A detailed description of each dataset used in the experiments is provided below.

RNA Data It is a classification data set, that consists of random extraction of gene expression of patients having five–different types of cancerous tumor: KIRC, PRAD, BRCA, LUAD and COAD [Fio16]. The dimensions of this data set are (801×20531). The number of attributes (20531) are significantly more than the number of instances (801).

GISETTE Data It is a handwritten digit recognition problem [Guy+04]. The task is to differentiate between highly confusible digits '4' and '9'. This data set is one of five data sets of the NIPS 2003 feature selection challenge. It is also a classification data set having dimensions of (6000×5000).

SPAM Data It is a textual data set that classifies emails as Spam or Non–Spam [Hop02]. It consists of 4457 instances which are pre–processed using TF–IDF method that quantifies the relevance of a text using statistical measures. Therefore, when TF–IDF approach is applied on SPAM data set the resultant data has (4457×5055) dimensions. This data set is widely used in natural language processing tasks.

ADULT Data It is a census income dataset [BK96], which consists of numerical and categorical values, and the target column is income, which indicates whether an individual's annual income exceeds 50K/yr. It is a classification data set which consists of 48000 instances and 14 attributes.

MADELON It is an artificially created dataset that consists of two–class classification problem with continuous input variables [Guy04]. It was a part of NIPS 2003 feature challenge having dimension of (4400×500).

Breast Cancer Data Breast cancer stands as the most prevalent form of cancer among women worldwide. This dataset is downloaded from [Kag], and sourced from the AI for Development organization, comprises information from 569 individuals, with each individual characterized by 31 features.

MNIST Data MNIST is a widely used database of handwritten digits commonly employed in image processing tasks [Ten]. It comprises a collection of ($60000, 28 \times 28$) images depicting digits ranging from 0 to 9. The objective is to cluster a given image of a handwritten digit into one of ten classes representing integer values from 0 to 9.

These datasets were carefully selected to enable broad experimentation and to evaluate whether the proposed methodologies are well–suited for various

Algorithm 6 M–LLEMDAV

Require: Data points $X = \{x_1, x_2, \ldots, x_N\}$ in $\mathbb{R}^D$

Ensure: Anonymized lower-dimensional representation $Y' = \{y_1, y_2, \ldots, y_N\}$ in $\mathbb{R}^d$ where $d < D$

1: Create a weighted graph M by connecting points x_i and x_j if their Euclidean distance $dist_E(x_i, x_j) \leq \epsilon$. Set the edge weights $w_{ij} = dist_E(x_i, x_j)$.

2: Calculate geodesic distance between points x_i and its neighbors that are selected in above step using Dijkstra shortest path algorithm.

3: Construct each point from its neighbours. Reconstruction errors are calculated by minimizing the cost function

$$\epsilon(w) = \sum_i |x_i - \sum_j w_{ij} x_j|^2$$

subject to constraint $\sum_{j=1}^{N} w_{ij} = 1$. Thus, weights w_{ij} are obtained that reconstructs each data point from its neighbors.

4: Compute the low–dimensional data Y that best preserves the manifold structure, represented by weights w_{ij}.

$$\phi(y) = \sum_i |y_i - \sum_j w_{ij} y_j|^2$$

subject to constraint $\sum_{i=1}^{N} y_i = 0$. Thus, lower–dimensional matrix $Y(N \times d)$ is resulted.

5: Apply M–MDAV to Y to perform k–anonymity and obtain the final anonymized dataset Y'.

types of real–world and artificially generated datasets. This diverse selection ensures a comprehensive analysis of the methods' effectiveness across different data modalities and structures. In the first approach, the M–MDAV algorithm directly anonymizes the high–dimensional data to achieve k–anonymity. Alternatively, M–ISOMDAV and M–LLEMDAV transform the data into a lower–dimensional space using ISOMAP and LLE, respectively, to preserve the manifold structure, followed by anonymization with M–MDAV. To evaluate performance, state–of–the–art machine learning algorithms, including SVM, Naive Bayes, Gradient Boosting, Decision Trees, Random Forests, XGBoost, and KNN, are implemented, and the best–performing model is identified for testing. Finally, utility is validated by recording evaluation metrics such as accuracy, precision, recall, and K–Stress, where K–Stress is defined as follows:

K–Stress is a weighted sum of differences between the distance in the original space, and their corresponding representations in the lower–dimensional space [Kar+05]. It is a measure of goodness of fit that requires that distance between two points in perturbed lower–dimensional embedding are well preserved with respect to distance between those points in original higher–dimensional

Table 3.1: Empirical Results of k–Anonymous Manifold Learning Approaches

Dataset	X(N,D)	Algorithm	Accuracy	Precision	Recall	K–Stress
RNA	800 × 20531	M–ISOMDAV	**99.17**	**99.18**	**99.17**	**0.43**
		M–LLEMDAV	58.12	59.3	58.13	0.73
		M–MDAV	90.10	90.12	90.11	–
Gisette	6000 × 5000	M–ISOMDAV	77.79	76.82	77.78	0.69
		M–LLEMDAV	**85.13**	**86.10**	**85.14**	**0.64**
		M–MDAV	69.21	69.87	69.18	–
SPAM	5272 × 5055	M–ISOMDAV	**85.20**	**84.34**	**85.21**	**0.45**
		M–LLEMDAV	42.61	43.13	42.59	0.89
		M–MDAV	39.56	40.10	39.81	–

space. The stress indicates the amount of information loss before and after transformation, and expressed as a percentage with 0% stress being equivalent to perfect transformation. Mathematically, it is calculated as follows:

$$\sqrt{\sum (d_{ij} - \delta_{ij})^2 / \sum d_{ij}^2} \tag{3.1}$$

where d_{ij} is the pairwise distance between points in higher–dimensional embedding, whereas δ_{ij} is the pairwise distance between points in lower–dimensional space. The K–Stress metric is not applicable to M–MDAV algorithm because it evaluates the preservation of pairwise distances between high–dimensional data points and their corresponding low–dimensional embeddings. Recall, M–MDAV operates entirely within the high–dimensional space and does not involve any transformation or mapping to a lower–dimensional space. Consequently, K–Stress cannot be computed for this approach, as there is no embedding process to assess for distance preservation.

Table 3.1 presents the empirical results obtained from the datasets using the three proposed approaches as discussed, providing a comparative analysis of their performance. The first column lists the names of the datasets, while the second column specifies their dimension in terms of the number of instances and attributes. The third column indicates the algorithm applied, and the subsequent columns detail the evaluation metrics, including accuracy, precision, recall, and K–Stress. For each dataset, the best–performing approach is highlighted in bold to clearly showcase the most effective methodology.

For k–anonymity, the parameter k was chosen after several iterations with different values. When k was set between 5 and 10, the outcomes in terms of accuracy were consistently good. However, increasing k to a range of 15–20 led to a decline in performance. As a k–value larger than 5 is commonly considered

acceptable for k–anonymity and micro–aggregation, we selected $k = 10$ as a generalized value for our experiments.

We utilized seven machine learning classification models for our analysis of three proposed approaches. Upon evaluation, the K–Nearest Neighbors (KNN) classifier emerged as the best–performing model for the RNA dataset. For testing and performance evaluation, we set the number of neighbors to 5 and the weight distribution to uniform. Conversely, for the Gisette and SPAM datasets, the Gradient Boosting Classifier performed best and was used for further evaluation. The chosen hyper–parameters for Gradient Boosting were: 100 estimators, a learning rate of 0.1, and a maximum tree depth of 5, while other parameters were kept at their default settings as provided by the scikit–learn library [Ped+11] in Python.

Upon analysis, it is observed that the M–ISOMDAV approach outperforms other methodologies for the RNA and SPAM datasets, achieving the highest accuracies of 99.17% and 85.20%, respectively. For the RNA dataset, the K–Stress value of 0.43 obtained using M–ISOMDAV is significantly better compared to the 0.73 achieved by M–LLEMDAV. Conversely, for the GISETTE dataset, the M–LLEMDAV approach provides the best results with an accuracy of 85.13% and a K–Stress value of 0.64. Notably, the M–MDAV approach fails to deliver optimal results for any of the datasets, with its performance significantly lagging behind the other two methods. This highlights that M–MDAV alone cannot effectively anonymize high–dimensional data while preserving its manifold structure, emphasizing the critical role of manifold learning in such scenarios.

The relatively poor performance of M–LLEMDAV on RNA and SPAM datasets can likely be attributed to the presence of multiple manifolds in these datasets. The LLE manifold learning algorithm, which utilizes various tangent linear patches to approximate a manifold, is better suited for simpler datasets like GISETTE. Its design, which relies on modeling a single manifold as multiple small linear functions, limits its ability to generalize effectively for datasets with complex manifold structures. This analysis underscores the need for manifold learning techniques tailored to the intricacies of the data to achieve optimal results in machine learning applications.

We also performed our experiments on other two datasets i.e., Adult and Madelon datasets which are presented in Table 3.2. We found that in the case of Adult and Madelon data set, the data points are not really in high–dimensions, as it should be for the manifold learning techniques. Also, the data–distribution for these datasets is not similar to the manifold structure. Thus, poor performance in terms of accuracy and neighborhood preservation (K–Stress) is obtained. We propose the following hypothesis based on the analysis of our results.

Hypothesis 1. *The data–points should really be in high–dimensions and must possess manifold structure, then only the proposed approaches will be able to learn the intrinsic structure of the manifold and anonymize data–points efficiently.*

Table 3.2: Datasets and Scenarios Where Proposed Approaches Showed Suboptimal Performance

Dataset	X(N,D)	Algorithm	Accuracy	Precision	Recall	K–Stress
Adult	48842×14	M–ISOMDAV	50.12	50.13	50.11	0.35
		M–LLEMDAV	43.32	43.30	42.29	0.32
		M–MDAV	41.19	42.90	40.12	–
Madelon	4400×500	M–ISOMDAV	62.18	62.25	62.19	0.28
		M–LLEMDAV	59.23	59.21	59.23	0.25
		M–MDAV	60.38	60.30	61.21	–

Based on the results in Table 3.1 and Table 3.2, we observed that k–anonymity alone is insufficient to ensure both strong privacy protection and high data utility in high–dimensional datasets with complex geometric structures. While differential privacy offers rigorous privacy guarantees, it often requires the injection of significant noise particularly problematic in scenarios involving manifold–based data representations, where the preservation of fine–grained local structures is critical. To address these limitations, we propose a hybrid anonymization technique that combines the strengths of k–anonymity with differential privacy. This hybrid approach enables a more effective balance between privacy protection and analytical utility. However, anonymization should not be evaluated solely through the lens of downstream machine learning performance. In many scenarios, the primary goal is to preserve and understand the underlying structure of high–dimensional data itself, especially when these structures encode meaningful patterns or behaviors.

To this end, we emphasize the importance of capturing the geometric and statistical properties of the data manifold, independent of any specific predictive task. In high–dimensional spaces, classical metrics often fail to reflect the true layout of the data. As a solution, we propose the use of the Fréchet mean as a robust, geometry–aware metric that better reflects the intrinsic data distributions and offers a meaningful way to measure structural fidelity after anonymization. This perspective broadens the scope of privacy–preserving data analysis–shifting from task–specific evaluation to a more foundational assessment of how well the anonymized data retains its high–dimensional characteristics.

3.2 Fréchet Mean

Statistical summaries, such as the mean, provide valuable insights about a dataset. The mean intuitively represents the central tendency of the data and is one of the most widely used measures of central tendency [Man11]. It

depicts critical information about the distribution, location, and structure of the dataset. However, for high–dimensional datasets with manifold structures, traditional measures like the arithmetic or geometric mean are insufficient to capture the inherent properties of the data. In such scenarios, the Fréchet mean [GK73], [Fré] offers a more suitable alternative. The Fréchet mean generalizes the concept of centroids to any metric space, making it capable of preserving the intrinsic geometry and structure of high–dimensional data. It produces a representative point for a cluster of points. For real numbers, finding a representative point p works by using Euclidean distance. In contrast, for a metric space $(\mathbb{M}, \delta)$, operations such as Fréchet mean are preferable.

Definition 5. *(Fréchet Mean) Let $(\mathbb{M}, \delta)$ be a complete metric space, and $X = \{x_1, x_2..., x_N\}$ be a data set with points in $\mathbb{M}$. We define the Fréchet mean as the point Z, that globally minimizes the objective function:*

$$Z = \arg\min_{p \in \mathbb{M}} \sum_{i=1}^{N} \delta(p, x_i)^2$$

As we have seen in Chapter 2, the sensitivity of a function provides an upper bound on how much data must be perturbed to preserve privacy. Reimherr et al. [RBS21] have studied the sensitivity of Fréchet mean on manifolds using the geodesic distance.

Theorem 1. *Let Z and Z' be the Fréchet mean of two databases DB_1 and DB_2. And, let $\rho(Z, Z')$ denote their manifold distance. Then it can be proven that, for all Z and Z' Fréchet means of databases DB_1 and DB_2 that only differ in one record, it holds the following*

$$\rho(Z, Z') \leq \frac{2D(2 - h(D, \kappa))}{Nh(D, \kappa)} \quad \text{where}$$

$$h(D, \kappa) = \begin{cases} 2D\sqrt{\kappa}cot(\sqrt{\kappa}2D) & if\ \kappa > 0 \\ 1 & if \kappa \leq 0 \end{cases} \tag{3.2}$$

Here, h is a function of D and κ derived from the Hessian comparison theorem (Theorem 11.7) in [Lee18], D is the length of records, N is the sample size, and κ is an upper bound of sectional curvature of $\mathbb{M}$. That is, the global sensitivity of the Fréchet mean is bounded by the above expression. We leverage this sensitivity to anonymize the Fréchet mean under the differential privacy framework, as well as in our proposed hybrid anonymization method.

3.3 k–Anonymity meets Differential Privacy

The primary motivation behind anonymizing the data that resides on a manifold is to preserve both its geometric structure and the essential information it contains. Conventional data privacy models operate either in high–dimensional

ambient space or in its low–dimensional embedding, transforming the data into a linear space where traditional privacy techniques can be applied. However, as observed in our experiments, applying k–anonymity to the low–dimensional representations obtained via ISOMAP and LLE manifold learning techniques led to poor generalization and information loss when the datasets are not in high–dimensions.

On the other hand, if we apply differential privacy mechanism directly on the high–dimensional space, its sensitivity calculations are based solely on the dimensions of this space. This results in overly conservative noise addition, significantly degrading the data utility and affecting downstream tasks [Kam+19]. To overcome these limitations, we propose: (β, k, ϵ_0)–anonymization method, which integrates k–anonymity with DP to achieve effective anonymization while preserving the intrinsic manifold structure of the data.

We provide a rigorous theoretical foundation for our method, demonstrating its robust privacy guarantees through formal proofs and analytical justifications. Furthermore, we empirically validate its effectiveness by comparing it against conventional k–anonymity and DP–based methods. Our experimental results highlight the superiority of our approach in terms of privacy protection, data utility, scalability, and performance in real–world scenarios. This dual validation–both theoretical and empirical–reinforces the credibility and practicality of our approach in ensuring privacy while maintaining the structural integrity and usability of high–dimensional data.

3.3.1 (β,k,ϵ_0)–anonymization method

We propose the (β, k, ϵ_0)–anonymization method, which provides formal privacy guarantees with carefully chosen values of β, k, and ϵ_0. This approach introduces a novel way to satisfy both differential privacy and k–anonymity simultaneously. Traditional differential privacy techniques often require adding a significant amount of noise to obscure sensitive information, particularly when dealing with high–sensitive queries. In contrast, our algorithm adopts an alternative approach: instead of relying on excessive noise, we introduce the sampling step combined with generalization at the initial stage of data processing. One well–known approach to enhance the privacy of a mechanism is to apply it to a random subsample of the input database rather than the entire dataset. This reduces the risk of leaking information about any particular individual, as no information can be revealed when the individual is not a part of the subsample. This structured transformation reduces the overall sensitivity of the data, thereby minimizing the magnitude of noise required to ensure DP compliance. The advantage of this algorithm lies in the fact that the error introduced in generalization step is likely to be more than compensated by the reduction in the noise required to attain DP, compared to the noise that would be required to attain DP with original data.

Definition 6. *Given a dataset X of N points, the subsample mechanism selects a random sample from the uniform distribution over all subsets of X of size*

m. The ratio $\beta = \frac{m}{N}$ *is defined as the sampling parameter of the subsample mechanism.*

We will need the following lemmas.

Lemma 1. *[Ull17] Let s represent a subsample mechanism with ratio* β*. Let M be a mechanism which is* (ϵ, δ)*–DP. Then the mechanism* $M' = M \circ s$ *is* (ϵ', δ')*–DP with* $\epsilon' = \beta\epsilon$ *and* $\delta' = \beta\delta$*.*

In mathematical notation, the composition of functions is often denoted by the symbol $\circ$. Therefore, the notation $M \circ s$ signifies applying the function M to the output of the subsample. Intuitively, the lemma says that subsampling with probability β <1 improves a (ϵ, δ)–DP algorithm to a $(\beta\epsilon, \beta\delta)$–DP algorithm for any ϵ and δ. Privacy budget reduction by the subsampling principle ensures that a differentially private mechanism run on a random subsample of a population provides higher privacy guarantees than when run on the entire population [SD17].

Let us now consider the sensitivity associated to clustering and centroids. As the contribution of a record to the centroid is inversely proportional to the cardinality of its corresponding cluster, the sensitivity of the centroid can be calculated by dividing the sensitivity of the record by the cluster's cardinality. This concept can be expressed formally through the following lemma.

Lemma 2. *[SD17] Let* $C \subset X$ *be a cluster of records in a dataset X and let* $\bar{C}$ *be the mean of the records in C. Let* ΔD *be the* L_1*–sensitivity of a record in the dataset X. The* L_1*–sensitivity of the centroid* $\bar{C}$ *is* $\Delta\bar{C} = \frac{\Delta D}{|C|}$*.*

Using all these preliminaries, we can propose our algorithm and provide its privacy guarantees.

Algorithm

Let c be the number of clusters, let k be the number of records in a cluster and assume $c = \lfloor |X|/k \rfloor$. The Algorithm 7 describes the step–by–step procedure of our proposed (β,k,ϵ_0)–anonymization method.

Theorem and Proof

Theorem 2. *(*β*,k,*ϵ_0*)* ***–anonymization algorithm****: Random sampling with probability* β *when 0 <*β *<1 followed by microaggregation of records into c clusters each with atleast k records and laplacian noise addition with scale*

$$b = \frac{\Delta D}{\epsilon_0 \cdot k}$$

satisfies ϵ*–differential privacy with* $\epsilon = \beta\epsilon_0$*.*

Algorithm 7 (β,k,ϵ_0)–anonymization

Require: Dataset X, parameters β, k, ϵ_0, *steps*
Ensure: Anonymized Dataset X'
1: **Initialize:** set of parameters β, k, ϵ_0
2: **for** `int i` $\leftarrow$ `1,...,` *steps* **do**
3: Draw a random sample X_s with prob β from X.
4: Micro–aggregate X_s in k–clusters.
5: Compute ΔD =L1–sensitivity of a record in X_s
6: Add Lap(0,b) into k–clusters with b = $\frac{\Delta D}{\epsilon_0 k}$
7: **end for**
8: Return Dataset X'.

Proof. Note that the microaggregation will produce c clusters. Let us consider that we protect each of them independently with ϵ_0–DP, then, the overall microaggregation will still be ϵ_0–DP. Each cluster has atleast k records. Therefore, according to Lemma 2, the sensitivity of a cluster is $\Delta D/k$, where ΔD is the sensitivity of one record. Therefore, we can achieve ϵ_0–DP for microaggregation with a $Lap(0, b)$ where $b = \Delta D/(\epsilon_0 k)$. This is precisely the parameter b used in the algorithm. Therefore, our approach of combining microaggregation and Laplacian noise produces ϵ_0–differential privacy.

The algorithm concatenates sampling and the differential privacy version of microaggregation. Therefore, we can apply Lemma 1. We have already seen that with the selected b leads to a ϵ_0–differential privacy mechanism. Or, equivalently $(\epsilon_0, 0)$–differential privacy. The sampling is with parameter β. Therefore, the application of Lemma 1 implies $(\epsilon_0\beta, 0\beta)$–differential privacy. So, in overall, the mechanism is ϵ= $\epsilon_0\beta$–differential privacy with $b = \Delta D/(\epsilon_0 \cdot k)$. Note that ϵ_0 is the parameter of the last iteration of microaggregation+ DP, while ϵ is the privacy budget of the whole (β,k,ϵ_0)- anonymization method. Therefore, if we want to apply (ϵ, δ)–DP, then we need to select β, k, ϵ_0 so that $\epsilon_0 = \epsilon/\beta$.

This completes the proof. □

Theorem 3. *(β,k,ϵ_0) –**anonymization algorithm**: Random sampling with probability β when $0 < \beta < 1$ followed by microaggregation of records into c clusters each with atleast k records and laplacian noise addition with scale*

$$b = \frac{\Delta D}{\epsilon_0 \cdot k}$$

satisfies k–anonymity.

Proof. Theorem by construction. □

Thus, our proposed anonymization method satisfies the formal definitions of both k–anonymity and differential privacy. To evaluate the effectiveness

of the proposed method beyond conventional downstream machine learning tasks, we instead adopted the Fréchet mean as a metric, which offers a geometric assessment of the data's intrinsic structure in high–dimensional spaces. We performed data anonymization using three different techniques: our proposed (β,k,ϵ_0)–anonymization method, as well as traditional methods such as k–anonymity and DP. For the k–anonymized Fréchet mean, we employed the M–MDAV algorithm as outlined in Algorithm 4 in the previous section. Once the anonymized dataset was obtained using Algorithm 4, the Fréchet mean was computed on this anonymized dataset. To generate the differentially private Fréchet mean, we added Laplace noise to the dataset based on the sensitivity bounds provided in Theorem 1 and the chosen value of ϵ. After noise addition, we computed the Fréchet mean on the differentially private dataset. To evaluate the effectiveness of our approach, we calculated the manifold distance between the original Fréchet mean (computed in the high–dimensional space) and the anonymized Fréchet mean obtained using all three approaches. This analysis demonstrates how closely the anonymized Fréchet mean aligns with the original mean. A smaller distance indicates better utility, as it shows that the anonymized data retains more of the original structure and information.

We also introduce a machine learning clustering model specifically tailored for high–dimensional dataset, termed Fréchet Mean Clustering. Unlike conventional clustering techniques that rely on the arithmetic mean (e.g., k–means), our model leverages the Fréchet mean to more accurately capture the underlying geometric structure of high–dimensional data. This approach is particularly beneficial when the data distribution deviates from simple linear assumptions, as is often the case in real–world high–dimensional datasets.

By adopting the Fréchet mean as the central tendency measure, we demonstrate that for high–dimensional spaces, it provides a more representative and geometry–aware alternative to the arithmetic mean. Consequently, Fréchet Mean Clustering serves as a meaningful downstream evaluation task, offering a more principled way to assess the structural preservation and utility of anonymized data. This shift allows us to move beyond standard machine learning tasks, and instead evaluate how well an anonymization method retains the intrinsic properties of the data manifold providing a stronger justification for using Fréchet mean–based metrics and models in privacy–preserving data analysis.

3.4 Fréchet Mean Clustering

Fréchet mean–based K–Means clustering introduces a novel approach to clustering datasets by utilizing the concept of the Fréchet mean to improve the robustness and accuracy of cluster representations. The algorithm starts by randomly selecting c data points from the dataset as initial cluster centroids, similar to traditional K–Means. Subsequently, each data point is assigned to the nearest centroid based on the Riemannian distance, as described in [NK17].

Algorithm 8 Fréchet Mean Clustering

Require: $X = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{M}$: dataset on a manifold $\mathbb{M}$, c: no. of clusters, τ_m: step size for Fréchet mean
Ensure: $V = \{v_1, v_2, \ldots, v_c\}$: Fréchet mean centroids, labels $= \{f_1, f_2, ..f_n\}$: cluster assignment for each x_i
1: **Initialize:** Choose c initial centroids $V = \{v_1, v_2, ..v_c\}$ randomly from $\mathbb{M}$
2: **repeat**
3: **for** $i = 1$ to N **do** ▷ d_{reim}: Riemannian distance
4: Assign x_i to closest cluster: $f_i \leftarrow \arg\min_{j \in \{1,\ldots,c\}} d_{reim}(x_i, v_j)$
5: **end for**
6: **for** $j = 1$ to c **do**
7: Update centroid v_j: $v_j \leftarrow \text{FréchetMean}(\{x_i \mid f_i = j\}, \tau_m)$
8: **end for**
9: **until** Centroids V converge when changes are below threshold
10: **return** V, labels

The Riemannian distance between two points (z_0, z_1) is defined as:

$$d_{reim}(z_0, z_1) = \frac{1}{2} \operatorname{arcosh}\left(1 + \frac{|z_1 - z_0|^2}{(1 - |z_0^2|)(1 - |z_1^2|)}\right) \tag{3.3}$$

This metric provides a suitable approach for clustering in non–Euclidean spaces. After assigning all data points to their respective clusters, the centroids are recalculated as the Fréchet mean of all data points within each cluster. The process of assigning data points to the nearest centroid and updating the centroids is iteratively repeated until convergence, which is typically defined by minimal changes in centroid positions or a set maximum number of iterations. This method offers a more refined measure of central tendency, particularly advantageous for datasets with complex structures or non–linearities. As a result, it leads to more meaningful and accurate clustering outcomes. The step–by–step explanation of the algorithm is provided in Algorithm 8.

3.5 Study Design

In this section, we present a step by step explanation of the complete workflow of our methodology.

1. Pre–processing: The first step involves normalizing the dataset to remove biases and inconsistencies, ensuring data quality and consistency for further processing.

2. Anonymization: The pre–processed data is anonymized with our proposed (β,k,ϵ_0)–anonymization method. To provide a comparative analysis we also apply traditional privacy models, including k–anonymity using

M–MDAV algorithm as outlined in Algorithm 4, and a differential privacy model.

3. Fréchet Mean Computation: The Fréchet mean is calculated on the anonymized dataset, obtained from the previous step. The Fréchet mean effectively captures the complex geometric structure of high–dimensional data, providing a robust central tendency measure in non–Euclidean spaces.

4. Measure the Manifold Distance: Compute the manifold (geodesic) distance, as defined in Definition 4 in Chapter 2, between the anonymized Fréchet mean (computed in the previous step) and the original Fréchet mean (computed on the pre–processed dataset). Since the dataset is anonymized using three different methods, we assess utility separately for each approach. This distance metric plays a critical role in evaluating the utility of anonymized datasets, particularly in high–dimensional and sparsely distributed data spaces. By quantifying the geometric discrepancies between the original and anonymized datasets, it provides key insights into the extent to which the underlying data structure is preserved. A smaller manifold distance indicates higher utility, signifying that the anonymization technique better retains the intrinsic properties of the dataset.

5. Fréchet Mean Clustering: Another approach to assess the utility of the anonymized dataset is through machine learning analysis. Traditional k–means clustering relies on Euclidean distance, which may not accurately capture relationships in high–dimensional, non–Euclidean spaces. In contrast, Fréchet mean–based clustering is better suited for such scenarios, as it computes cluster centroids as Fréchet means, which provide a more geometrically meaningful representation of the average point within each cluster. This method enhances the interpretability and representativeness of cluster centroids, particularly in high–dimensional spaces. In the final step, clustering performance is evaluated using Normalized Mutual Information (NMI) and the Silhouette Score, which measure the quality and coherence of the clustering assignments.

3.6 Experimental Results and Discussion

The experiments are conducted on diverse datasets, including RNA dataset, MNIST and Breast Cancer dataset. We analyze the results from various perspectives including the effect of sample size, the impact of ϵ and k on manifold distances, with a comparison between them. Additionally, we assess machine learning performance through Fréchet mean clustering, determining the optimal number of clusters using the elbow method. A detailed discussion of the results is provided in the following sections.

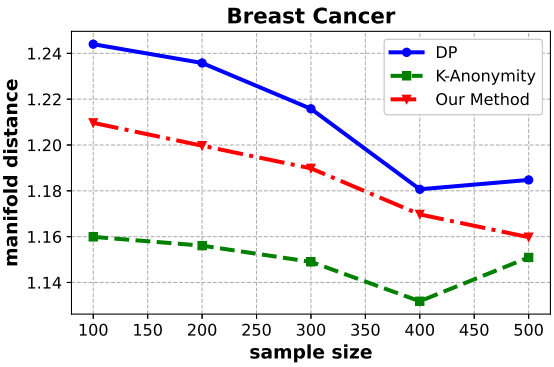
3.6.1 The Effect of Sample Size on Manifold Distance

The objective of this analysis is to examine how the sample size influences the Fréchet mean of the data. Intuitively, the larger the sample size, the more accurate is the mean of the data. As the sample size increases, the dispersion of the data gets smaller, and the mean of the distribution becomes closer to the population mean. To investigate this, we computed the Fréchet mean on both the original pre–processed dataset (denoted as Z) and the anonymized dataset (denoted as Z') obtained through three different anonymization methodologies: our proposed (β, k, ϵ_0)–anonymization method, as well as k–anonymity and Differential Privacy models. We then calculated the manifold (geodesic) distance between them, denoted as $\rho(Z, Z')$, as defined in Definition 4.

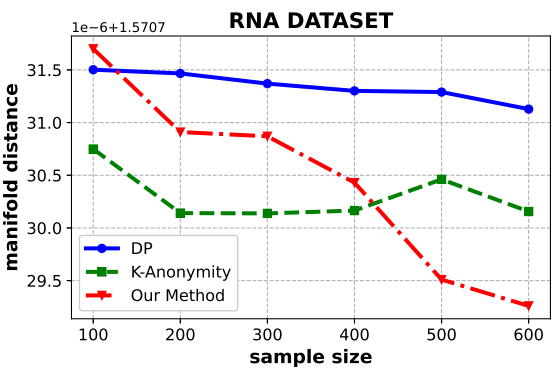
In Figure 3.1, the x–axis represents the sample size, where samples are randomly drawn from the entire dataset using a uniform distribution. The y–axis shows the manifold distance $\rho(.,.)$ calculated between the original Fréchet mean Z and the anonymized Fréchet mean Z'. The Fréchet mean Z is computed by minimizing the objective function directly on the high–dimensional data X, as described in Definition 5. On the other hand, the anonymized Fréchet mean Z' is derived by anonymizing the dataset using the three models: k–anonymity, Differential Privacy, and our proposed (β, k, ϵ_0)–anonymization method outlined in Section 3. In the plot, the blue line represents the Differential Privacy model, the green line corresponds to the k–anonymity method, and the red line illustrates the manifold distance computed using our proposed approach. As expected, manifold distance decreases as the sample size increases, a trend observed across all approaches for each dataset. Specifically, Figure 3.1 reveals that the Breast Cancer and RNA datasets show similar patterns, where the manifold distance between the original and anonymized Fréchet means is closer when using the k–anonymity model. However, for the MNIST dataset, we observed a different trend, where the Differential Privacy model outperformed the k–anonymity model. In these cases, the DP model more effectively captures the structure and information of the data compared to the k–anonymity method. Furthermore, when combining both k–anonymity and DP through our proposed (β, k, ϵ_0)–anonymization method, the resulting manifold distance plot lies between those of the k–anonymity and DP models. This suggests that our approach offers a better utility to privacy trade–off than either model individually. By blending elements from traditional models with novel methodological considerations, our approach preserves more of the data's geometric structure, reducing the loss of information typically associated with anonymization. This analysis underscores the effectiveness of our proposed method in maintaining data utility while ensuring privacy protection.

3.6.2 The impact of ϵ of DP on manifold distance

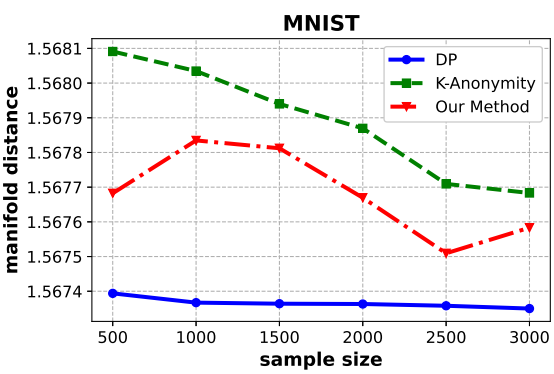
We analyze the impact of the Differential Privacy parameter ϵ on the manifold distance while keeping the sample size fixed. The ϵ parameter plays a crucial



(a) Breast Cancer



(b) RNA



(c) MNIST

Figure 3.1: Sample size vs Manifold distance (ρ) for different datasets. The blue line represents anonymized results using DP, the green line represents anonymized results using k–anonymity, and the red line represents anonymized results using our method.

role in DP by controlling the level of noise introduced to anonymize the data. A lower ϵ value enforces stronger privacy by adding more noise, whereas a higher ϵ value reduces noise, thereby improving data utility. However, the optimal value of ϵ cannot be determined a priori and must be empirically evaluated based on its effect on the results. To systematically examine this trade–off, we conducted experiments across a range of ϵ values, from 0.01 to 1.00.

Figure 3.2 (a), (c), (e) illustrate the relationship between ϵ and the manifold distance between the original and anonymized Fréchet means. As expected, increasing ϵ leads to a reduction in manifold distance, indicating that the anonymized mean becomes closer to the original mean. This is because a higher ϵ value introduces less perturbation, thereby retaining more of the dataset's geometric structure. This trend is consistently observed across all datasets, reinforcing the fundamental trade–off between privacy and accuracy in DP.

To determine an appropriate ϵ value for each dataset, we identify the point at which the manifold distance stabilizes, suggesting diminishing gains in utility despite further increases in ϵ. For the Breast Cancer dataset, this occurs around $\epsilon = 0.04$, where the curve transitions into a near–linear trend. Similarly, for both the RNA and MNIST datasets, the optimal ϵ value is found to be approximately 0.03 based on the same criterion. These values provide a balance between privacy preservation and data utility, ensuring that the anonymization process retains meaningful structural information while minimizing information leakage.

3.6.3 The impact of k of k–Anonymity on manifold distance

We analyze the impact of the parameter k in the k–Anonymity privacy model on the manifold distance while maintaining a fixed sample size. The parameter k determines the level of anonymity by ensuring that each record in the dataset is indistinguishable from at least $k-1$ other records. As k increases, privacy is strengthened because more records are grouped together and replaced by identical values, reducing the granularity of the data. However, this increased privacy comes at the cost of utility, leading to a larger distortion in the dataset. Consequently, the manifold distance between the original and anonymized Fréchet means increases, indicating a greater deviation from the original data distribution.

Figure 3.2 (b), (d), (f) illustrate this inverse relationship between k and data utility across different datasets. As k increases from 5 to 40, the privacy level improves at the expense of utility, resulting in a corresponding rise in the manifold distance between the original and the anonymized Fréchet mean. This trend is consistently observed across all datasets, including Breast Cancer, RNA, and MNIST, reinforcing the trade–off between privacy and data utility. It is particularly noteworthy that for smaller values of k, the increase in privacy introduces minimal distortion, leading to relatively low manifold

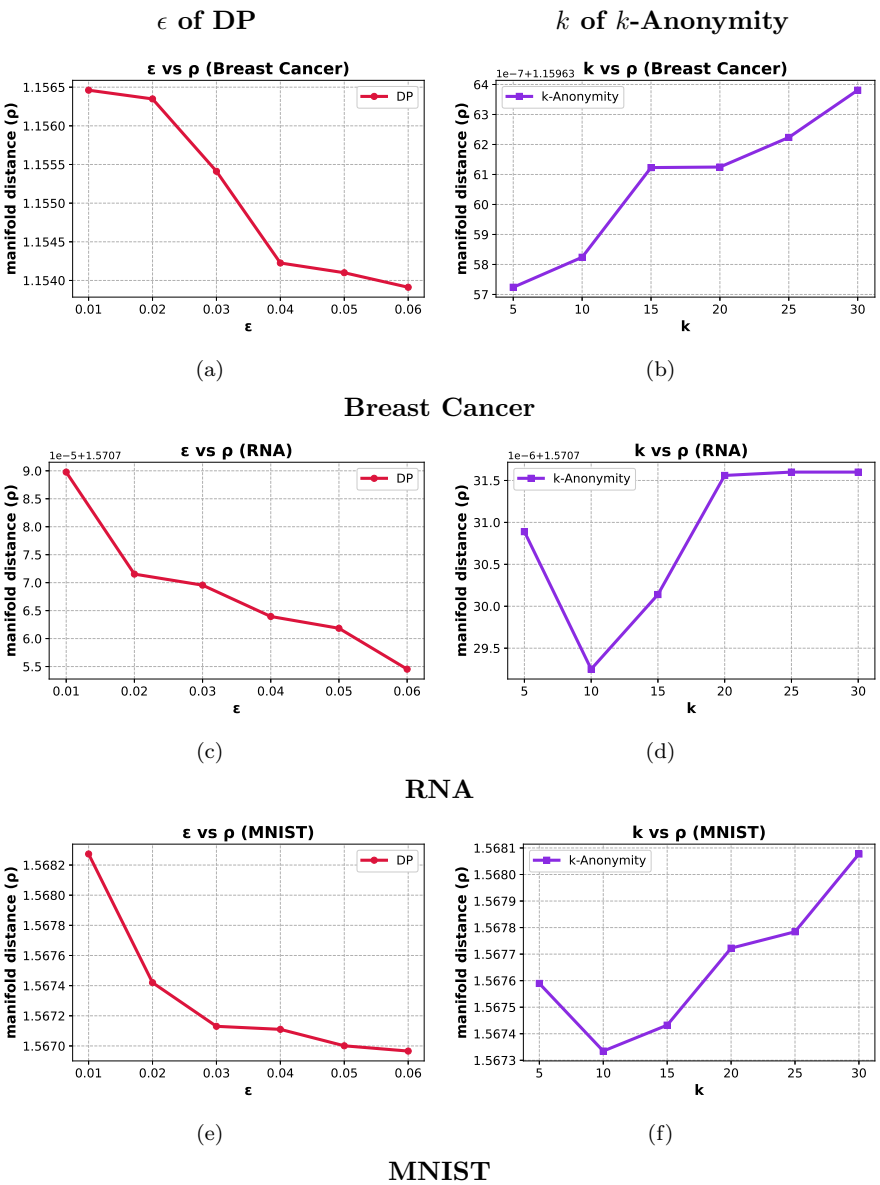


Figure 3.2: Epsilon of Differential Privacy and k of k-Anonymity vs Manifold Distance (ρ) for a fixed sample size across different datasets: Breast Cancer (569×31), RNA (801×20531), and MNIST ($60000 \times (28 \times 28)$).

distances. This observation aligns with existing findings in machine learning literature, where slight perturbations in data do not significantly degrade model performance [AY04]. Machine learning models often exhibit robustness to small modifications in input data, meaning that minor levels of anonymization may not substantially impact model accuracy. However, as k continues to grow, the excessive generalization of data results in a more substantial loss of structure, thereby increasing the discrepancy between the original and anonymized data representations in the manifold space. Selecting an optimal k value thus requires balancing privacy requirements with utility constraints, ensuring that data remains useful for downstream tasks while complying with anonymity standards.

3.6.4 Comparison between ϵ and k

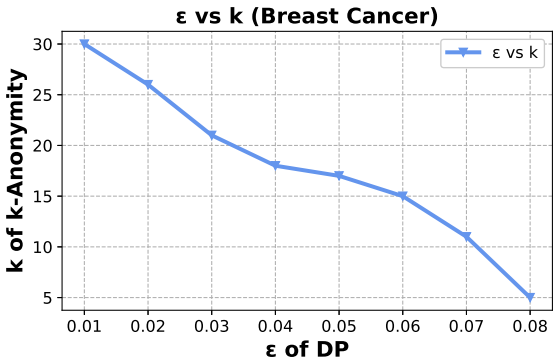
We also analyzed the relationship between the Differential Privacy parameter ϵ and the k value in k–Anonymity by identifying pairs (ϵ, k) that yield comparable manifold distances. In other words, we examined combinations where the manifold distance between the original Fréchet mean (Z) and the anonymized Fréchet mean (Z') remains similar under both anonymization techniques. Figure 3.3 illustrates these equivalent levels of perturbation across different datasets.

For instance, in the Breast Cancer dataset, we observed that when $k = 10$, the resulting manifold distance is equivalent to that obtained with $\epsilon = 0.07$. Similarly, in the MNIST dataset, a k value of 10 corresponds to $\epsilon = 0.071$, producing a comparable level of perturbation. However, this relationship is dataset–dependent. For example, when $k = 15$, the equivalent ϵ values differ significantly: in the Breast Cancer dataset, $\epsilon = 0.06$, while in the RNA dataset, it is $\epsilon = 0.035$, and in the MNIST dataset, it drops further to $\epsilon = 0.026$.

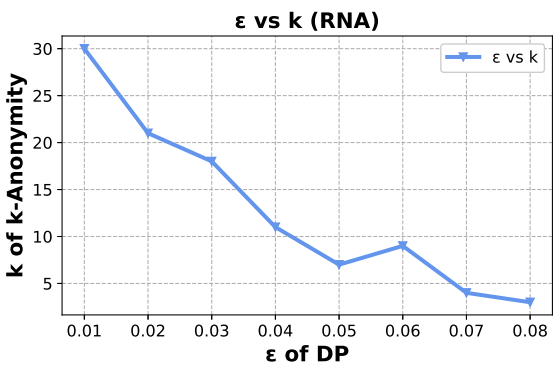
This discrepancy arises from the inherent differences in dataset distributions and sample sizes. Specifically, the MNIST dataset contains a larger number of unique values, enabling Differential Privacy to achieve comparable utility at a lower ϵ value, thereby providing stronger privacy guarantees. This observation aligns with the theoretical underpinnings of Differential Privacy, which suggest that as sample size increases, the mechanism can introduce noise more effectively while maintaining data utility.

A clear inverse relationship between ϵ and k emerges from our findings. As ϵ values increases, k values decrease, reflecting the fundamental trade–off between privacy and utility. A lower k value indicates less generalization in k–Anonymity, preserving more granular data utility but reducing privacy. Conversely, a lower ϵ value implies stronger privacy in Differential Privacy at the cost of increased noise, potentially reducing utility. The highest level of privacy is achieved when k is maximized and ϵ is minimized, ensuring both anonymization techniques offer the strongest possible protection.

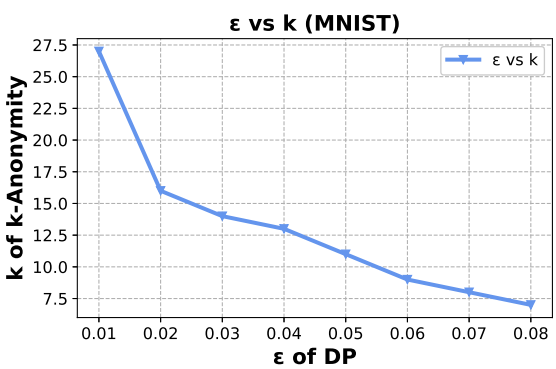
These findings confirm that both k–Anonymity and Differential Privacy can be effective in preserving manifold structures under appropriate parameter se-



(a) Breast Cancer

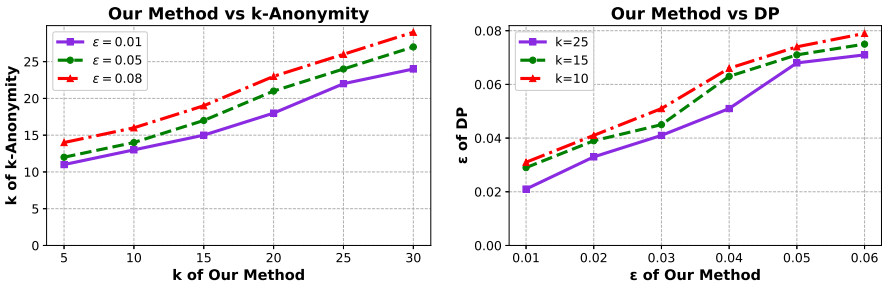


(b) RNA



(c) MNIST

Figure 3.3: Relationship between different privacy models: ϵ of Differential privacy vs k of k–Anonymity for 3 data sets, where blue line depicts a fixed distortion in terms of manifold distance (ρ) between original and anonymized Fréchet mean.



(a) k–Anonymity (b) DP

Figure 3.4: Comparison of Privacy Models: (β,k,ϵ_0)–anonymization vs k–Anonymity and DP

lection. However, choosing a single privacy model that consistently outperforms across diverse datasets and applications remains challenging. This motivates our proposed (β, k, ϵ_0)–anonymization method, which seeks to leverage the strengths of both models. Instead of relying on a single technique, our approach integrates key aspects of k–Anonymity and Differential Privacy, striking a balance between privacy and utility. By combining their benefits, (β, k, ϵ_0)–anonymization enhances data utility while preserving the geometric structure of high–dimensional data, making it a more robust and adaptable solution for privacy–preserving data anonymization across various contexts.

3.6.5 Comparison of Privacy Models

We introduced a novel methodology for the comparative evaluation of different privacy–preserving techniques, addressing the complexity introduced by multiple parameters in our proposed (β, k, ϵ_0)–anonymization method. Unlike traditional models such as k–Anonymity and ϵ–Differential Privacy, which rely on a single parameter, our method incorporates multiple factors to achieve a more balanced trade–off between privacy and utility.

To ensure a meaningful comparison, we systematically analyzed the relationship between the k parameter in k–Anonymity and its counterpart in our (β, k, ϵ_0)–anonymization method while maintaining a fixed ϵ value. Our results indicate that for a given ϵ, the manifold distances obtained using our method and traditional k–Anonymity exhibit a consistent trend across varying k values. Specifically, the k parameter in (β, k, ϵ_0)–anonymization aligns closely with the k value in standard k–Anonymity in terms of its impact on manifold distances. This trend is visually depicted in Figure 3.4(a), where the x–axis represents the k values in our (β, k, ϵ_0)–anonymization method, and the y–axis corresponds to those in traditional k–Anonymity. The plotted lines indicate different fixed ϵ values (e.g., 0.01, 0.05, and 0.08), demonstrating their respective influences on privacy preservation. Notably, lower ϵ values, such as 0.01, correspond to

stricter privacy guarantees, as expected from DP principles.

Furthermore, we performed a comparative analysis of the ϵ values in our (β, k, ϵ_0)–anonymization method and those in traditional ϵ–DP while keeping k constant. This relationship is illustrated in Figure 3.4(b), where we examine the impact of ϵ on privacy and utility trade–offs. Our findings indicate that for a fixed k, the manifold distances obtained under our method closely follow the trends observed in traditional DP, reinforcing the validity of our approach. Although these results were derived using the Breast Cancer dataset, the observed patterns are consistent across diverse datasets, highlighting the generalizability of our methodology.

By systematically analyzing the interplay between multiple privacy parameters, our approach provides a rigorous framework for evaluating privacy models beyond traditional single–parameter techniques. This contributes to advancing privacy–preserving data analytics by offering a more flexible and adaptable anonymization strategy that accounts for dataset characteristics and privacy requirements in a principled manner.

3.6.6 Assessing ML Performance with Fréchet Mean Clustering

Through the application of Fréchet mean clustering, which leverages the Fréchet mean for computing centroids and Riemannian distance for similarity measurements, we obtained compelling clustering results across multiple datasets. To evaluate the effectiveness of Fréchet mean clustering approach, we employed Normalized Mutual Information (NMI) and the Silhouette Score, two widely used metrics in clustering analysis. The NMI measures the agreement between clustering assignments and ground truth labels, with higher values indicating a stronger correspondence. The Silhouette Score quantifies the cohesion and separation of clusters, where higher values suggest well–defined and distinct clusters. By using these evaluation criteria, we ensured a rigorous assessment of clustering quality under different privacy–preserving transformations.

Our experimental results demonstrate that the (β, k, ϵ_0)–anonymization method consistently outperforms privacy models such as k–Anonymity and DP across all datasets, as presented in Table 3.3. Specifically, our method achieves the highest NMI values, indicating superior retention of structural information in the anonymized data. Furthermore, it also yields the highest Silhouette Scores, signifying well–separated and coherent clusters despite the anonymization process. These findings underscore the robustness of our approach in preserving meaningful data patterns while ensuring privacy.

Analyzing the results in more detail, we observe that k–Anonymity generally maintains better utility than DP, particularly in datasets with structured categorical attributes, such as the RNA and Breast Cancer datasets. However, the increase in k reduces the granularity of data, leading to a decline in clustering performance. On the other hand, DP introduces random noise, which impacts the data distribution and subsequently reduces clustering per-

Table 3.3: Clustering Analysis using NMI and Silhoutte score

Dataset	**Breast Cancer**		**RNA**		**MNIST**	
	NMI	Silhoutte score	NMI	Silhoutte score	NMI	Silhoutte score
k–Anonymity	0.44	0.30	0.61	0.22	0.50	0.07
DP	0.49	0.25	0.47	0.12	0.47	0.09
(β, k, ϵ_0)–anonymization	0.74	0.38	0.84	0.25	0.61	0.08

formance, especially on datasets with high–dimensional continuous attributes like MNIST. In contrast, our proposed method effectively balances privacy and utility, achieving a more optimal trade–off between protection and data integrity.

A key observation from our study is that the relative impact of different privacy models varies across datasets. For instance, in the Breast Cancer dataset, our method achieves an NMI of 0.74 and a Silhouette Score of 0.38, significantly surpassing both k–Anonymity (0.44, 0.30) and DP (0.49, 0.25). Similar trends are observed in the RNA dataset, where our approach attains an NMI of 0.84 and a Silhouette Score of 0.25, outperforming k–Anonymity (0.61, 0.22) and DP (0.47, 0.12). The MNIST dataset, being inherently more complex and high–dimensional, presents more challenges for privacy–preserving clustering. Nevertheless, our approach still outperforms the baseline models, achieving an NMI of 0.61 and a Silhouette Score of 0.08, compared to k–Anonymity (0.50, 0.07) and DP (0.47, 0.09).

These results indicate that our (β, k, ϵ_0)–anonymization method not only preserves privacy but also retains essential structural characteristics of the data, making it particularly effective for tasks requiring meaningful clustering. The findings reinforce the notion that traditional privacy models often involve trade–offs that may not be optimal across all datasets, whereas our approach provides a more adaptable and reliable solution. Our hybrid (β, k, ϵ_0)–anonymization method offers more flexibility than traditional models, as it combines the strengths of k–anonymity (for $\beta = 1$, k) and ϵ-differential privacy (for $\beta = 1$, $k = 1$). This allows us to select the optimal parameters from each model, though finding the best set of parameters can be challenging.

3.6.7 Determine suitable number of cluster with Elbow Method

The elbow method is a widely used technique in cluster analysis for determining the optimal number of clusters in a dataset. It involves plotting the inertia (also known as the within–cluster sum of squares) against the number of clusters. As the number of clusters increases, inertia always decreases since clusters become more refined and data points are grouped more specifically. However, beyond a certain point, adding more clusters results in diminishing returns, where the reduction in inertia is no longer significant. This inflection point, or elbow, marks the optimal number of clusters, striking a balance between maximizing

variance explained and avoiding over–fitting due to excessive complexity.

In Figure 3.5, we present the elbow plot, which depicts the relationship between the number of clusters (k) and inertia. The key principle behind the elbow method is identifying the point where the rate of decrease in inertia significantly slows down, forming an elbow–like shape in the graph. This plateauing effect suggests that additional clusters provide minimal improvement in cluster compactness, making the chosen k at the elbow the most suitable choice for the dataset.

Our analysis, applied across multiple datasets, confirms the effectiveness of this method in identifying meaningful cluster structures. For instance, in the Breast Cancer dataset, our results indicate that the optimal number of clusters is two, aligning with the underlying biological classification of tumors into malignant and benign groups. This outcome reinforces the validity of our clustering approach in medical diagnostics, where precise classification is crucial for decision–making.

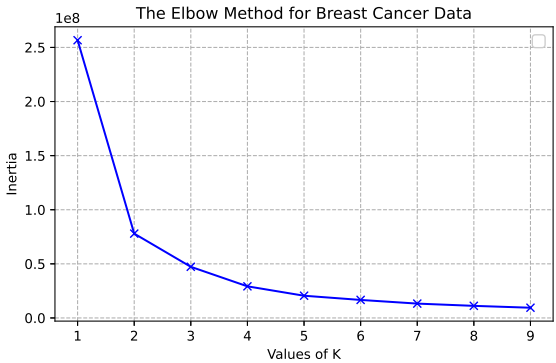
Similarly, in the RNA dataset, the inertia curve suggests that the ideal number of clusters could be five or six, as evidenced by the point where the decline in inertia slows down. This result is particularly relevant for analyzing genetic expression profiles, where multiple distinct patterns of gene expression can be uncovered through clustering. The ability to distinguish between these patterns is essential for biomedical research, disease classification, and personalized medicine applications.

For the MNIST dataset, a well–known benchmark in image recognition, we observe a similar trend. The optimal cluster count aligns with the inherent structure of the dataset, further demonstrating the reliability of our method. Given the high–dimensional nature of handwritten digit images, the fact that the elbow method successfully identifies an interpretable clustering structure underscores the robustness of our Fréchet mean clustering approach.

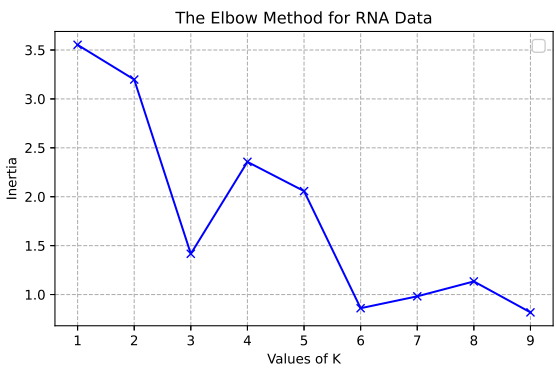
Overall, our findings validate the effectiveness of the proposed Fréchet mean clustering model, showcasing its ability to detect intrinsic data structures across diverse application domains, including medical diagnostics, genetic data analysis, and image processing. By leveraging Riemannian geometry in cluster analysis, our approach enhances interpretability and preserves the fundamental relationships within data, offering a superior alternative to traditional clustering techniques. These insights not only improve our understanding of complex datasets but also facilitate more data–driven decision–making in fields where privacy, accuracy, and structure preservation are paramount.

3.7 Conclusion

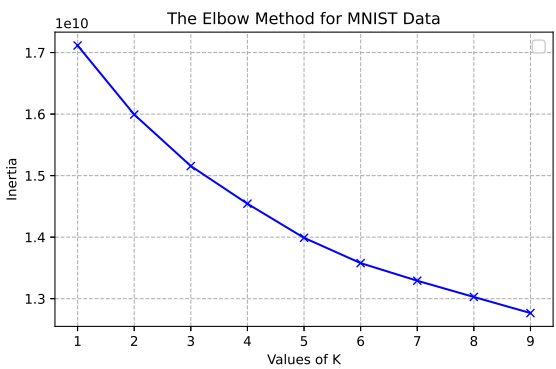
In this chapter, we explored techniques for anonymizing high–dimensional data while preserving its intrinsic structure. We began with the k–anonymity privacy model and extended it using manifold learning techniques such as ISOMAP and Locally Linear Embedding (LLE). These methods enabled us to develop pri-



(a) Breast Cancer



(b) RNA



(c) MNIST

Figure 3.5: Elbow Method to find number of clusters vs Inertia.

vacy–preserving models that anonymize high–dimensional data while maintaining its underlying manifold structure. To achieve this, we proposed novel approaches: M–MDAV, M–ISOMDAV, and M–LLEMDAV, which leverage manifold learning for effective anonymization. Our experiments revealed that these techniques are particularly effective when applied to genuinely high–dimensional datasets that exhibit a well–defined manifold structure. In such cases, the models successfully preserve data utility while achieving privacy protection. However, a critical challenge remains: accurately measuring information loss in high–dimensional spaces is a non–trivial task. Traditional evaluation metrics often fail to capture the complexity of transformations in such settings.

To address this issue, we introduced the concept of the Fréchet mean as a statistical measure for assessing information loss in high–dimensional spaces. The mean, as a fundamental statistic, encapsulates key properties of a dataset, making it an intuitive and powerful tool for evaluating privacy–preserving transformations. By leveraging the Fréchet mean, we provided a principled approach for quantifying distortions introduced by anonymization techniques. Furthermore, we highlighted the limitations of both k–anonymity and differential privacy in high–dimensional settings. Selecting a single privacy model that consistently outperforms others across diverse datasets and applications is challenging. To overcome this, we proposed the (β, k, ϵ_0)–anonymization method, a hybrid approach that combines the strengths of k–anonymity and DP, ensuring improved privacy guarantees while maintaining higher utility compared to traditional models. Our theoretical analysis and empirical validation demonstrated that (β, k, ϵ_0)–anonymization consistently outperforms standalone k–anonymity and DP, offering a more robust and adaptable privacy framework. However, due to the presence of multiple parameters in this (β, k, ϵ_0)–anonymization method, fine–tuning them effectively can be challenging and may require careful calibration to balance privacy and utility.

Finally, we extended our work by introducing Fréchet Mean Clustering, a machine learning model designed specifically for high–dimensional spaces. This approach enhances the applicability of the Fréchet mean in clustering tasks, further reinforcing its role in preserving data structure while enabling meaningful pattern discovery. Overall, our work provides a comprehensive framework for high–dimensional data anonymization, addressing both privacy and utility trade–offs. By integrating manifold learning, hybrid privacy models, and advanced clustering techniques, we contribute to the advancement of privacy–preserving data analysis in complex, high–dimensional domains.

Chapter 4

Beyond Anonymization: Synthetic Data Solutions

> Generative models are a key enabler of machine creativity, allowing machines to go beyond what they've seen before and create something new
>
> — *Ian Goodfellow*

Ensuring privacy in AI–driven systems is a critical challenge, particularly when dealing with sensitive, high–dimensional datasets. This chapter builds upon previous efforts in this thesis to develop privacy–aware AI systems, focusing on the role of synthetic data generation as an alternative to direct data anonymization. Traditional anonymization techniques, such as generalization and perturbation, often struggle to balance privacy and utility, especially when datasets are high–dimensional or subject to adversarial attacks. Synthetic data generation has emerged as a promising solution, that mimics the original dataset, aiming to retain the statistical properties and underlying relationships of the original dataset without directly exposing individual records. By design, synthetic data should prevent adversaries from reconstructing or linking specific individuals to their real–world counterparts, thereby enabling privacy–preserving data sharing across institutions and industries.

In high–dimensional scenarios, preserving geometric properties, structural integrity and the relative positioning of data points is crucial, as neglecting these can compromise utility. However, synthetic data is not inherently immune to privacy risks. Reconstruction attacks, membership inference attacks, and linkage attacks can still expose patterns that leak sensitive information, challenging the assumption that synthetic data is inherently safe. Therefore, it is essential to assess the privacy guarantees of synthetic data generators, par-

ticularly in high–dimensional settings where complex patterns are harder to obfuscate. This chapter addresses these challenges by analyzing existing synthetic data generation techniques, identifying their strengths and limitations, and proposing new approaches for generating high–quality, privacy–preserving synthetic data. This aligns with Research Question **RQ2**, which explores techniques for generating high–quality synthetic data, particularly in the context of high–dimensional real–world datasets.

In this chapter, we first propose a framework that replaces high–dimensional sensitive data with synthetic data generated using Generative Adversarial Networks (GANs). The goal is to create synthetic datasets that closely resemble the original data while preserving privacy. We then explore whether incorporating prior knowledge about the dataset during GAN training can enhance the quality of the generated data. In addition to evaluating various synthetic data generators, we also focus on understanding their distributional learning capabilities. To achieve this, we visualize the generated data, assessing how well the models capture the underlying structure of the real dataset. We now turn our attention to the development of synthetic data generators specifically designed for high–dimensional data. This section focuses on preserving the geometric properties of the synthetic data, ensuring that its structure and relationships are maintained while still achieving privacy preservation.

4.1 The Need of Synthetic Data Generators for High–Dimensional Data

The objective is to develop high–quality synthetic data generation approach, specifically for high–dimensional real–world datasets. Synthetic data generation has been a prominent research area for the past two decades. However, early research primarily focused on generating artificial images, while structured data such as tabular, time–series, and categorical datasets have only recently gained attention. Among the various generative models designed for tabular data, CTGAN, proposed by Xu et al. [Xu+19], is one of the most widely used. CTGAN effectively models multi–modal distributions in continuous variables and mitigates class imbalance in discrete variables, enabling the synthesis of realistic tabular data suitable for analytical tasks. While several GAN architectures–such as Vanilla GAN and WGAN–have been explored for tabular synthetic data generation, CTGAN stands out due to its compatibility with structured datasets. However, GANs inherently learn from training samples, which raises significant privacy concerns [HAP17]. To mitigate this issue, several privacy–preserving synthetic data generators have been proposed. For instance, DPGAN [Xie+18] ensures privacy guarantees through differential privacy mechanisms, while PATEGAN [YJS19] employs a teacher–student architecture to bound the privacy risks. ADSGAN [YDV20] provides a legal and ethical solution for data sharing. Despite their effectiveness in addressing privacy concerns, these models do not explicitly focus on preserving the intrinsic

structure of high–dimensional data, which is crucial for generating high–utility synthetic data.

Beyond GANs, Variational Autoencoders (VAEs) have also been widely adopted for synthetic data generation. Akrami et al. [Akr+20] proposed RTVAE, a model designed to handle both categorical and continuous features while being robust to outliers. However, RTVAE does not incorporate privacy–preserving mechanisms, making it susceptible to potential privacy risks. Most existing synthetic data generation methods focus either on generating synthetic data for different kinds or focuses on privacy preservation while overlooking the importance of manifold learning–which is crucial for modeling complex, high–dimensional data structures [Sná+17]. In high–dimensional spaces, data typically resides on a low–dimensional manifold, meaning that traditional distance metrics (such as Euclidean distance) fail to capture the true relationships between data points [Dok+15]. This leads to distortions in the synthetic data, which significantly compromise its utility for downstream analytical tasks.

Manifold learning techniques, such as t–SNE [VH08] and UMAP provide an effective way to map high–dimensional data into lower–dimensional representations while preserving the intrinsic structure. By integrating these techniques into the synthetic data generation process, we ensure that the underlying geometric relationships of the original dataset are retained. This enhances the utility of the generated data for machine learning tasks, statistical analysis, and real–world decision–making. We now explore how we can generate high–quality synthetic data for high–dimensional data that replaces sensitive data.

4.2 Generate Privacy–Preserving Synthetic Data using M–KCTGAN Approach

In this section, we describe our proposed methodology, M–KCTGAN, for generating synthetic data while preserving both privacy and the manifold structure of the original dataset. The flowchart of the proposed algorithm can be visualized in Figure 4.1. Our approach begins with a high–dimensional real dataset $X(N, D)$, where N represents the number of samples and D denotes the number of features (Step 1 in Figure 4.1). To effectively capture the underlying manifold structure, we employ t–SNE, to transform $X(N, D)$ into a lower–dimensional representation $Y(N, d)$, where $d < D$. The goal is to train a CTGAN model on $Y(N, d)$ to generate synthetic data that retains the original manifold properties.

However, GANs are prone to privacy risks, as they can inadvertently memorize and leak sensitive training data. Key concerns include the memorization of individual records [Web+19] and membership inference attacks against generative models [Hay+17]. To mitigate these risks and prevent our model from overfitting to real data, we apply anonymization on $Y(N, d)$ (Step 2 in Figure 4.1). This is achieved using M–MDAV, the manifold–aware k–anonymization technique, which is detailed in Algorithm 4. Once anonymization is performed, we

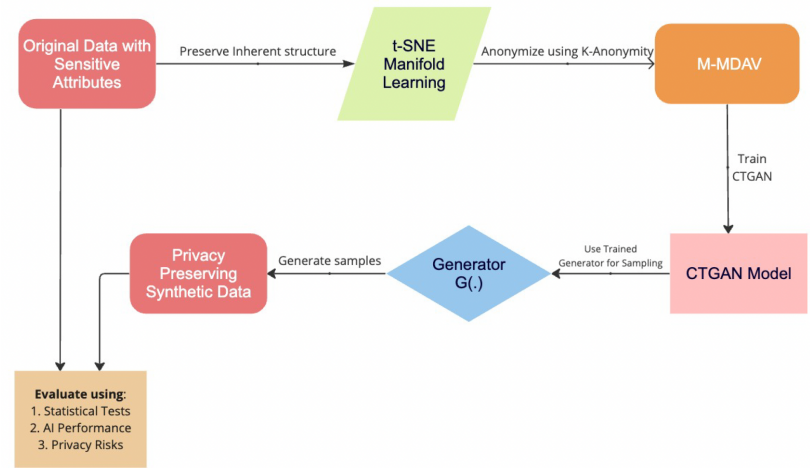


Figure 4.1: Privacy Preserving Synthetic Data Generation using M–KCTGAN Approach

train our CTGAN model on the anonymized dataset (Step 3 in Figure 4.1). The trained generator is then used to produce synthetic samples that preserve the underlying manifold properties (Steps 4 and 5 in Figure 4.1). Finally, to facilitate a meaningful comparison between the synthetic and real data distributions, we map the synthetically generated data back to the original high–dimensional space using a neural network. This transformation ensures that the real and synthetic datasets can be compared in the same feature space, making it easier to assess shape similarity and distribution alignment. A step–by–step breakdown of our proposed M–KCTGAN approach is provided in Algorithm 9.

Different manifold learning techniques, including ISOMAP, LLE, and UMAP, were empirically tested, and t–SNE was ultimately chosen for this approach. t–SNE is particularly effective when preserving fine local structures, which is essential for capturing small, tightly–knit clusters in GAN training. Unlike ISOMAP, which focuses on global geometry, or LLE, which may struggle with complex manifolds, t–SNE maintains meaningful local relationships while minimizing distortion. Although UMAP is efficient, t–SNE's probabilistic mapping ensures that similar points in the high–dimensional space remain close in the lower–dimensional representation, aiding the GAN in learning underlying patterns.

Since, t–SNE lacks a built–in inverse transformation to map low–dimensional embeddings back to the original high–dimensional space, we employed a neural network (autoencoder) to learn the reverse mapping. t–SNE's non–linear embedding is particularly effective for datasets with complex relationships and the neural network offers a highly flexible and accurate reconstructions from low–dimensional t–SNE embeddings. Additionally, t–SNE's stochastic nature

Algorithm 9 M-KCTGAN

Require: Original Dataset $X = \{x_1, x_2, \ldots, x_N\}$ in $\mathbb{R}^D$
Ensure: $X_s^{'}(N, D)$ is manifold privacy preserving synthetically generated data

1: Convert Euclidean distance between high–dimensional data points (x_i and x_j) into conditional probabilities, representing similarities, where σ is the variance between data points.

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)} \tag{4.1}$$

2: Compute similar conditional probabilities for low–dimensional counterparts y_i, and y_j.

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)} \tag{4.2}$$

3: Minimize sum of differences of $p_{j|i}$ and $q_{j|i}$ using KL Divergence, and obtain $Y(N, d)$ where $d \ll D$.
4: Anonymize the data $Y(N, d)$ using M–MDAV and obtain protected records $Y'(N, d)$.
5: Train CTGAN model on protected records $Y'(N, d)$.
6: Generate privacy–preserving synthetic samples $Y_s'(N, d)$.
7: Transform synthetically generated data to its high–dimensional embedding using neural network and obtain $X_s^{'}(N, D)$.
8: Evaluate original records $X(N, D)$ and privacy–preserved synthetically generated records $X_s^{'}(N, D)$ using statistical and privacy metrics.

helps to handle noisy data by focusing on fine local structures, making it beneficial for synthetic data generation. Though UMAP supports approximate inverse mapping, its global optimization strategy may not always preserve fine local structures as effectively as t–SNE. The neural network based reverse transformations provide more precision and adaptability for capturing data nuances.

4.2.1 Emphasizing the Importance of Manifold Structure with a Comparison to the KCTGAN Approach

To emphasize the importance of preserving the manifold structure in our synthetic data generation process, we conducted a comparative analysis between our proposed approach (M–KCTGAN) and the KCTGAN algorithm. In the KCTGAN approach, instead of transforming the high–dimensional dataset into a low–dimensional space using t–SNE, the algorithm directly applies the M–MDAV Algorithm 4 on the original dataset $X(N, D)$. Following this, the CTGAN model is trained on the anonymized data to generate synthetic samples. This comparative analysis is essential for highlighting the advantages of incorporating manifold learning techniques in our methodology. By leveraging

Algorithm 10 KCTGAN

Require: Original Dataset $X = \{x_1, x_2, \ldots, x_N\}$ in $\mathbb{R}^D$
Ensure: $X'_s(N, D)$ is privacy preserving synthetically generated data
1: Anonymize the data $X(N, D)$ using M–MDAV k–anonymity privacy model (Algorithm 4 and obtain privacy–preserved records $X'(N, D)$.
2: Train CTGAN model on protected records $X'(N, D)$.
3: Generate privacy–preserving synthetic samples $X'_s(N, D)$.
4: Evaluate original records $X(N, D)$ and privacy–preserved synthetically generated records $X'_s(N, D)$ using statistical and privacy metrics.

t–SNE to project the data into a low–dimensional space before anonymization using M–MDAV, we ensure the preservation of the intrinsic data structure. This process mitigates the risk of information loss and improves the quality of the generated synthetic data. For a more detailed understanding of the algorithmic steps, please refer to Algorithm 10.

4.3 Evaluation Metrics and Privacy Assessment

We conducted experiments using datasets similar to those discussed in previous chapters, including the RNA dataset, Gisette dataset, and Adult dataset. In this section, we outline the evaluation metrics employed to assess both statistical and machine learning performance. Additionally, we discuss the data reconstruction attack applied to the synthetically generated datasets to evaluate potential privacy risks and vulnerabilities.

4.3.1 Statistical Evaluation of Data Utility

We evaluate the utility of the synthetic data by assessing its statistical performance, specifically focusing on whether the synthetically generated data from our proposed framework can effectively preserve the distribution and correlations inherent in the real data. To quantify this, we employ two widely–used measures: the Maximum Mean Discrepancy (MMD) and the Fréchet Inception Distance (FID), both of which are robust two–sample tests that compare the statistical properties of real and synthetic datasets.

Maximum Mean Discrepancy (MMD). It calculates the dissimilarity between two probability distributions P_r and P_s using samples drawn independently from each distribution [FM53; Gre+12]. The larger the MMD statistic, the greater is the dissimilarity between the distributions. Mathematically, we compute the square of MMD as follows.

$$MMD^2 = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} K(x_i, x_j) - \frac{2}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} K(x_i, y_j) + \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} K(y_i, y_j) \tag{4.3}$$

Here, K is a kernel function. We used Gaussian RBF Kernel [STV04] to compute similarity between the joint data.

$$K(x, x') = exp(\frac{-||x - x'||^2}{2\sigma^2}) \tag{4.4}$$

Fréchet Inception Distance (FID). It embeds a set of generated samples into a feature space [Heu+17], and estimates the mean and covariance for both real and synthetic data. The Fréchet distance between two Gaussian distributions is equivalent to their Wasserstein–2 distance and quantifies the quality of generated synthetic samples as follows.

$$FID(r, s) = ||\mu_r - \mu_s||_2^2 + Tr(\Sigma_r + \Sigma_s - 2(\Sigma_r \Sigma_s)^{\frac{1}{2}}) \tag{4.5}$$

where (μ_r, Σ_r) and (μ_s, Σ_s) are the mean and covariances of real and synthetic data respectively, and Tr is the trace of the matrix. Lower FID depicts smaller distances between synthetic and real samples. It appears to be a good measure, even though it only considers the first two order moments of the distributions. It has been shown that FID is consistent with human judgements and is more robust to noise [Heu+17].

Although FID is generally used to compute the feature distance between real images and generated images as it is able to capture the structure, location and order of points in a curve. It can also be used to investigate if synthetically generated data preserve the inherent manifold structure of the data. This is the motivation behind using this metric to investigate if the synthetic data generated can also preserve the inherent structure of data with minimal loss.

4.3.2 ML Performance in Classification Tasks

Our methodology leverages synthetic data for training machine learning models, followed by testing on real–world samples. The classification of these samples is carried out using a variety of models, including Support Vector Machines, Decision Trees, and Gradient Boosting. To assess the performance of these models, we use the $F1$ score as the primary evaluation metric, focusing on identifying the model that delivers the best performance within our framework. The $F1$ score is a metric that combines precision and recall into a single value to evaluate the performance of a classification model. More precisely, it is the harmonic mean of precision and recall, providing a balanced measure when there is an uneven class distribution. The formula for the $F1$ score is:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.6}$$

where precision and recall are defined as:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN} \tag{4.7}$$

In these formulas, TP denotes true positives, FP denotes false positives, and FN denotes false negatives. The $F1$ score is used because it provides a balanced evaluation of a classification model's performance, especially in cases where the dataset has an imbalanced class distribution. In such scenarios, accuracy can be misleading, as a model may predict the majority class well but fail to identify the minority class. The $F1$ score combines precision (the ability to correctly identify positive samples) and recall (the ability to capture all positive samples), providing a balanced evaluation by accounting for both false positives and false negatives.

4.3.3 Privacy Evaluation: Data Reconstruction Attack

The objective of generating synthetic data is to ensure that the statistical properties of the real dataset are well–preserved while preventing the retention of specific local characteristics that might expose sensitive information about individual records. This is crucial for safeguarding personally identifiable information (PII). We consider a scenario where an adversary has access to both real and synthetic datasets and attempts to establish a mapping between a real data point r_s and a synthetic data point s_s. To assess the privacy risks associated with synthetic data generation, we conduct the following steps. For each synthetic sample s_s, we identify its closest real counterpart r_s by computing the minimum Euclidean distance d_s between them. The overall privacy leakage is then estimated by computing the mean and variance of these minimal distances, i.e., across all synthetic samples s_s.

Formally, given a real data point $r_s \in \mathbb{R}^D$ and a synthetic sample $s_s \in \mathbb{R}^D$, we define a successful linkage for the synthetic sample s_{s_i} as the closest real sample r_{s_i} that lies closest than the other real sample r_{s_j}. We define an indicator function for this $\theta_{s_s} : [D] \to \{0, 1\}$, which determines whether a successful linkage has occurred:

$$\theta_{s_s}(i) = \begin{cases} 1, & \text{if } |s_{s_i} - r_{s_i}| = \min_{j \in [D]} |s_{s_i} - r_{s_j}| \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

The total number of successfully linked records is then defined as:

$$\text{link}(s_s) = \sum_{i \in [D]} \theta_{s_s}(i) \tag{4.9}$$

Additionally, a disclosure risk assessment is performed to evaluate whether the synthetic data generator effectively preserves privacy. A disclosure event occurs if an adversary can infer that a specific real data record was used in training by analyzing synthetic samples [Cho+17]. This type of attack is referred to as a data reconstruction attack. Formally, disclosure is considered to have occurred if a synthetic sample s_s is found within a predefined threshold distance ϵ from its closest real counterpart r_s.

4.4 Results and Discussion

A comprehensive analysis was conducted to evaluate both the statistical and ML performance of the synthetically generated data. Additionally, a privacy evaluation was performed to determine whether any sensitive information was inadvertently disclosed. A well–known challenge in GANs is that the density of the learned generative distribution tends to concentrate around the training data points, potentially leading to memorization of these samples due to the high model complexity of deep neural networks. To address this, we compared our proposed approach with several baseline privacy–preserving models, including DPGAN, ADSGAN, and PATEGAN. We also compared our method with RTVAE to assess their ability to generate synthetic data while maintaining the manifold structure of the original data. RTVAE was selected due to its effectiveness in synthetic data generation for tabular data and its support for β–divergence, which allows flexible control over the trade-off between reconstruction accuracy and latent space regularization.

4.4.1 Utility Evaluation

Table 4.1 presents the statistical performance metrics, comparing the original datasets with the synthetically generated datasets using MMD and FID. Both MMD and FID metrics are designed to assess the dissimilarity between synthetic and real data distributions, with lower values indicating higher similarity. These metrics are particularly useful for evaluating how well the synthetically generated data captures the underlying manifold structure of the real data. Our proposed approach, M–KCTGAN, yields the lowest MMD and FID values across all three datasets when compared with other privacy–preserving approaches. The integration of the t–SNE manifold learning technique in our method aids in capturing the most relevant data information while filtering out potential noise, ensuring that the data is represented in a more informative and low–dimensional space. As a result, synthetic data generated using M–KCTGAN exhibit a distribution closer to that of the original data, while preserving privacy.

Among the baseline generative models, CTGAN demonstrates superior performance in modeling structured data, achieving lower MMD and FID values compared to other baselines such as DPGAN, PATEGAN and RTVAE. This

Table 4.1: Statistical Evaluation: MMD and FID score

Dataset	ADULT		GISETTE		RNA	
	MMD	FID	MMD	FID	MMD	FID
RTVAE	1.14×10^{-3}	1.85×10^{9}	3.97×10^{-4}	1.84×10^{8}	3.12×10^{-3}	3.21
DPGAN	1.10×10^{-3}	2.44×10^{11}	4.47×10^{-4}	7.23×10^{8}	2.51×10^{-3}	1.43×10^{5}
ADSGAN	1.10×10^{-3}	9.05×10^{8}	4.79×10^{-4}	7.33×10^{8}	3.12×10^{-3}	- 4.05
PATEGAN	1.15×10^{-3}	1.94×10^{10}	4.99×10^{-4}	7.18×10^{7}	3.16×10^{-3}	- 3.63
CTGAN	1.24×10^{-3}	1.18×10^{9}	4.15×10^{-4}	2.22×10^{8}	2.98×10^{-3}	2.95×10^{5}
KCTGAN	1.10×10^{-3}	2.29×10^{8}	3.99×10^{-4}	2.08×10^{8}	2.63×10^{-3}	6.21×10^{4}
M-KCTGAN	7.10×10^{-4}	1.95×10^{8}	3.41×10^{-4}	1.76×10^{8}	2.52×10^{-3}	3.59×10^{4}

Table 4.2: $F1$-score when trained on synthetic data and tested on real data

Method	**ADULT**	**Gisette**	**RNA**
RTVAE	0.510	0.53	0.22
DPGAN	0.086	0.55	0.23
ADSGAN	0.173	0.57	0.04
PATEGAN	0.035	0.54	0.13
CTGAN	0.025	0.58	0.39
KCTGAN	0.035	0.61	0.40
M-KCTGAN	0.591	0.70	0.51

highlights the effectiveness of CTGAN in handling the complexities of tabular data, particularly in learning multi–modal distributions for continuous attributes and in addressing class imbalances in categorical variables. Given its strong performance and widespread adoption, we chose to enhance CTGAN rather than other models, building on its advantages while addressing its limitations in preserving geometric properties and privacy.

To further assess the effectiveness of the synthetic data, we performed machine learning classification tasks in which a classification model was trained on the synthetic data and tested on the original dataset. The $F1$ scores presented in Table 4.2 demonstrate that our approach outperforms other methods, highlighting the effectiveness of M–KCTGAN in generating high–quality synthetic data.

Despite achieving a higher $F1$ score compared to existing methods, the overall performance remains suboptimal with respect to the real data. This could be attributed to several factors, one of which is the insufficient preservation of dependencies between attributes. If critical feature relationships present in the real data are not accurately captured in the synthetic data, models trained on synthetic data may fail to learn meaningful decision boundaries. To address this limitation, we will further investigate and refine our approach to enhance the modeling of attribute dependencies.

Table 4.3: Mean and Variance of minimal distance between real and synthetic nearest neighbors

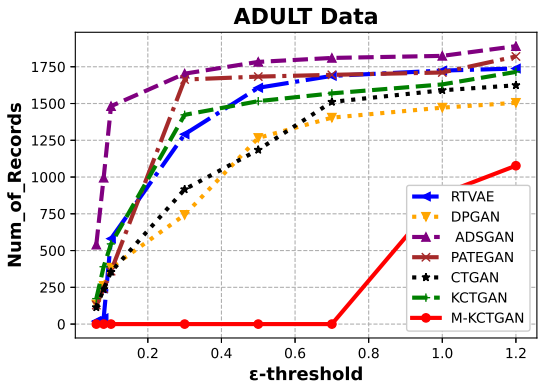
Method	**ADULT (48842 × 14)**	**Gisette (6000 × 5000)**	**RNA (800 × 20531)**
RTVAE	1672 $\pm 15 \times 10^6$	20324 $\pm 13\times$ 9.47	74 ± 3.47
DPGAN	25036 $\pm 13 \times 10^7$	26610 $\pm 13\times$ 10^3	190 $\pm 28\times$ 10^{-2}
ADSGAN	536 $\pm 81 \times 10^5$	26609 $\pm 12\times$ 10^3	192$\pm 23 \times 10^{-2}$
PATEGAN	4711 $\pm 19 \times 10^7$	13926 $\pm 24\times$ 10^5	91± 54
CTGAN	3605 $\pm 85 \times 10^7$	22955 $\pm 80\times$ 10^4	780$\pm 10^4$
KCTGAN	1552 $\pm 26 \times 10^5$	22547 $\pm 45\times$ 10^4	56 ± 32
M–KCTGAN	1126 ± 47	21620 ± 850	33 ± 26

4.4.2 Privacy Risk Evaluation

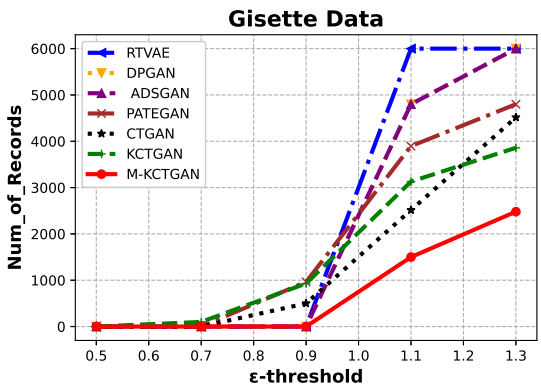
To assess the privacy risks of synthetically generated samples, we performed an analysis to measure the proximity between each synthetic sample and its closest neighbor in the real data. This was done by calculating the minimum Euclidean distance between synthetic samples and real data, while also considering the mean and variance of the distances, as detailed in Table 4.3.

For the Adult dataset, we observed that the ADSGAN model yields a minimum mean of 536, with a variance of 81×10^5. In contrast, M–KCTGAN produced a mean of 1126 and a much lower variance of 47. Although the ADSGAN model achieved a lower minimum mean, it resulted in a significantly higher variance. To further investigate, we computed the mean and variance of the minimal distances within the original real data samples themselves. Interestingly, we found that these values were similar to those of the synthetic data generated by ADSGAN. The Adult dataset consists of various categorical variables, which were encoded using ordinal encoding during the pre–processing phase. Due to the diverse types of variables, the data points are quite scattered, leading to a high variance in the distance measures. However, when the M–KCTGAN approach was applied, the t–SNE effectively retained only the most relevant information while filtering out noise. This approach contributed to a more compact representation of the data, resulting in a lower variance in the distances. Similar trends were also observed for the other two datasets, demonstrating the ability of M–KCTGAN to reduce variance while preserving privacy.

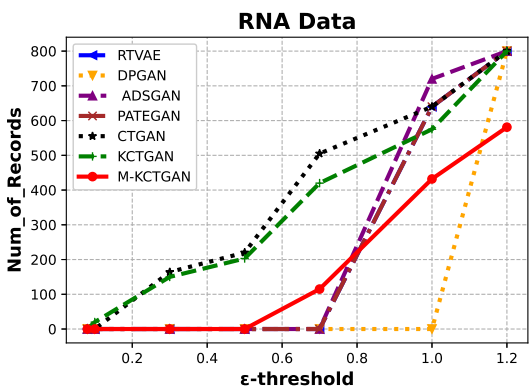
We also performed a data reconstruction attack to examine whether a real sample was involved in training the model and if privacy leakage had occurred. To do this, we verify whether the synthetic sample falls within a certain threshold distance from the original sample. We plot the graph threshold value (computed in terms of mean) vs number of records. When the minimal Euclidean distance is such that $d_s < \epsilon$, we consider that disclosure has occurred. We count the number of records whose privacy is compromised. This is plotted in Figure 4.2. The number of records is cumulative, as it increases with an increase of the threshold. For the Adult dataset, after a certain threshold value, for instance $\epsilon > 0.6$, the number of records does not increase significantly. How-



(a) Adult Dataset



(b) Gisette Dataset



(c) RNA Dataset

Figure 4.2: Data Reconstruction Attacks: Plot of ϵ threshold vs Number of Generated Record

ever, when data reconstruction attack is performed using the M–KCTGAN approach, no disclosure has taken place for ϵ upto 0.7. Afterwards also, i.e., when $\epsilon > 0.8$, the number of records that are disclosed is quite less than for the other two approaches. Thus, we can say that the proposed approach M–KTCGAN is least vulnerable to data reconstruction attack and also able to preserve the manifold structure of a high–dimensional data. Even for RNA data, the results from M–KCTGAN seems the best until $\epsilon = 0.5$, and progressing until $\epsilon = 0.7$, which is almost the best except for DPGAN.

4.5 Challenges with Tabular Data

Generating synthetic data using GANs for tabular datasets, such as the Adult dataset, presents significant challenges compared to other datasets like Gisette. The primary difficulty lies in the high number of categorical variables within tabular data, as GANs are typically more effective with continuous variables. While GANs have demonstrated considerable success with continuous data, they face substantial limitations when applied to tabular datasets, which often contain discrete attributes. Tabular datasets, which are common in fields such as healthcare, finance, and social sciences, have unique statistical properties that make them difficult to model using traditional GAN architectures. These datasets exhibit heterogeneous data types, imbalanced distributions, and complex dependencies between variables, all of which add layers of complexity that GANs struggle to capture. One of the reasons we achieved the suboptimal $F1$ score in Table 4.2 is the difficulty of the GANs to learn the dependencies between attributes. Unlike image or text data, which possess inherent spatial or sequential structures, tabular data lacks these easily recognizable patterns, and its relationships are often high–dimensional and intricate.

One of the primary obstacles in applying GANs to tabular data is the non–differentiable nature of discrete attributes, which limits the gradient–based optimization typically used in GANs. Several approaches have been proposed to address this issue. For instance, [KH16; Che+17] introduced differential models that incorporate specialized functions to handle discrete data, while [Yu+17] employed reinforcement learning to train non–differentiable models, particularly for natural language generation. Additionally, convolutional neural networks [Par+18] and recurrent neural networks [XV18] have been adapted to learn the marginal distributions of columns in tabular data. To overcome these difficulties, specialized GAN architectures have been proposed, such as CTGAN (which we have already seen) and CTAB–GAN [Zha+21], designed specifically for tabular data. These models address some of the challenges by focusing on the generation of discrete values and the preservation of dependencies between variables. However, they are still constrained by their reliance on fixed assumptions about the structure of the data and remain sensitive to issues like training instability. Despite advancing the field of tabular data generation, these models fall short of fully capturing the complexity of real–world data distributions,

as noted by [MS24].

A crucial challenge that remains is the integration of domain knowledge and prior statistical information to enhance the fidelity of synthetic tabular data. Current GAN–based models primarily rely on learning patterns from raw data, often neglecting essential domain–specific relationships that could improve data quality and utility. Incorporating structured prior knowledge into GAN architectures could significantly improve the robustness of generated data and mitigate existing limitations. To address this, we utilize Bayesian Networks to encode prior knowledge about the dataset into GANs, allowing for a more informed and structured approach to tabular data generation.

4.5.1 Bayesian Network

A Bayesian Network (BN) [CH92] is a probabilistic graphical model representing the joint probability distribution of a set of random variables using a directed acyclic graph (DAG). Each node corresponds to a random variable, and directed edges represent conditional dependencies. The joint probability distribution is factorized as

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \text{Parents}(X_i)) \tag{4.10}$$

where $\text{Parents}(X_i)$ are the parent nodes of X_i. There are several methods to learn the structure of BN from the data. Constraint–based methods (e.g., the PC Algorithm [KB07]) use statistical tests to identify independencies between variables. Score–based methods (e.g., Hill Climbing (HC) [TBA06]) evaluate the quality of a network structure using a scoring criterion, such as the Bayesian Information Criterion (BIC). In this work, we chose HC because of its efficiency in identifying probabilistic dependencies in large datasets.

The HC algorithm starts with an empty graph and iteratively adds, removes, or reverses edges between nodes. Each modification is evaluated using a scoring function like the BIC, which balances model complexity and data likelihood. The algorithm continues making improvements until it converges on the best network structure. The dependencies in the graph capture the conditional relationships between variables and serve as valuable auxiliary information. Specifically, each variable in the network has a set of parent nodes, which represent the variables that directly influence it. We used the pgmpy python package [AP15] to construct the BN structure as described.

4.5.2 Datasets Description

In this analysis, we specifically worked with different discrete tabular datasets. A prominent example of such datasets is social science data, which typically contains a higher proportion of categorical attributes. We used three distinct social science datasets in our experiments. The first is the Adult dataset [BK96],

Table 4.4: Description of Tabular Datasets

Dataset	# of Instances	# of Categorical Attr.	# of Numerical Attr.
ADULT	48842	9	6
SD2011	5000	21	14
Credit Risk	1000	6	4

which is a pre–processed version of the 1994 US Census data collected from over 45,000 individuals. This well–known dataset has been used in various ML experiments, including those in our prior research in Chapter 3. The second dataset is the Social Diagnosis 2011 (SD2011) [JT11], which focuses on defining both objective and subjective measures of quality of life in Poland. It is a raw census dataset containing 35 attributes, predominantly categorical, with key variables such as education level, smoking status, work experience abroad, and duration spent abroad. We chose SD2011 due to its realistic challenges, such as missing values, outliers, and messy entries, making it a more representative choice compared to cleaner or simulated datasets. This ensures that our experiments address the complexities typically encountered in real–world, minimally pre–processed data. The third dataset used is the German Credit Risk dataset [Hof94], which classifies individuals as either low or high credit risks based on various attributes, including savings amount, checking amount, credit amount, and credit history. The characteristics of each dataset, including the number of instances and attributes, are summarized in Table 4.4. These datasets were selected as representative examples of discrete social science data, with the objective of incorporating their inherent structures and prior knowledge into GANs for improved synthetic data generation.

4.6 Integrating Prior Knowledge into GANs

We focus on improving the quality of synthetic tabular data by proposing three distinct approaches to incorporate prior knowledge into GANs, enhancing their ability to generate high–quality, realistic data. First, we integrate public knowledge as constraints in the adversarial loss function, introducing a penalty for violations. This ensures that the generated data adheres more closely to the known patterns, thereby improving its fidelity and realism. Second, we enforce the preservation of the original data's correlation structure. This step is critical for maintaining statistical consistency between the synthetic and real data, ensuring that relationships between variables are accurately represented. Third, we model attribute dependencies using a Bayesian network, encoding these dependencies as embeddings. These embeddings are then integrated into Conditional GANs (CGANs) [Mir14] to guide the data generation process, allowing the model to better capture complex dependencies among attributes. To address the privacy concern, we also incorporate differential privacy into our

GAN framework by adapting noise injection technique. These modifications strike a balance between privacy and utility, ensuring that the synthetic data remains useful for downstream tasks while providing robust privacy protection. These techniques are described in detail as follows.

4.6.1 Public Constraint GAN (PCGAN)

Real–world datasets often contain publicly known constraints, such as logical boundaries or dependencies between variables. Incorporating these constraints into GANs prevents the generation of implausible or unrealistic data, thereby improving the authenticity and utility of synthetic outputs. To achieve this, domain–specific constraints are embedded directly into the GAN training process as penalty terms within the generator's loss function. These constraints serve as additional guidance, ensuring that synthetic data adheres to predefined rules or logical relationships. Since they apply universally rather than being dataset–specific, they do not raise privacy concerns. For instance, human age can be restricted to a realistic range (0–120 years) by introducing a penalty term for values outside this range:

$$\text{Penalty}_{\text{age}} = \text{mean}\left(\max(0, -\text{age}) + \max(0, \text{age} - 120)\right) \tag{4.11}$$

These penalties are incorporated into the generator's total loss function:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \sum_{i \in I} \lambda_{\text{i}}\text{Penalty}_{\text{i}} \tag{4.12}$$

where I is the set of penalties, $\mathcal{L}_{\text{adv}}$ is the adversarial loss, and $\lambda_{\text{adv}}, \lambda_{\text{i}}$ are weighting coefficients. During training, penalty terms are computed and back-propagated alongside adversarial loss, ensuring that generated data respects real–world constraints while maintaining diversity.

Incorporating Specific Data Constraints

For the Adult dataset, we enforce an age constraint (0–120 years) with a penalty coefficient (λ_{age}) of 10, experimentally determined to balance constraint adherence while preserving data fidelity. For the SD2011 dataset, we apply three constraints: age constraint (similar to adult dataset), smoking constraint and work–abroad constraint. Smoking constraint ensures consistency between smoking status and the number of cigarettes smoked. If an individual is labeled as a non–smoker, their cigarette count should be zero:

$$\text{Penalty}_{\text{smoking}} = \text{mean}\left((\text{smoke} < 0.5) \cdot |\text{nociga}|\right) \tag{4.13}$$

where smoke represents smoking status (non–smokers encoded as values < 0.5) and nociga represents the number of cigarettes smoked. Work–abroad constraint ensures logical consistency between working abroad status and duration. If a person is marked as working abroad (workab > 0.5), their recorded duration (wkabdur) must be non–negative:

$$\text{Penalty}_{\text{wabroad}} = \frac{1}{n} \sum_{i=1}^{n} (\mathbb{I}(\text{workab}_i > 0.5) \cdot \max(0, -\text{wkabdur}_i)) \tag{4.14}$$

Each of these constraints is integrated into the generator's loss function with a penalty coefficient of 10. For the German Credit Risk dataset, we enforce two constraints: an age constraint and a purpose constraint. The purpose constraint applies penalties if the credit amount exceeds predefined thresholds for specific purposes, such as 5000€ for vacation or repairs and 15,000€ or 20,000€ for business or education. These thresholds were determined through dataset analysis and aligned with real–world expectations, ensuring the generated data remains realistic while maintaining diversity and utility.

4.6.2 Correlation Structure GAN (CSGAN)

An effective strategy for ensuring that synthetic data retains the structural properties of the original dataset is to align their correlation matrices, which encapsulate pairwise variable relationships. To achieve this, categorical variables are first numerically encoded using a Label Encoder [Ped+11]. The correlation matrix of the real dataset, denoted as C_{real}, is computed and serves as a reference. Similarly, the correlation matrix of the synthetic dataset, $C_{\text{synthetic}}$, is computed during training. Any discrepancies between these matrices are minimized by incorporating a penalty term into the loss function. The penalty is defined using the Frobenius norm, which quantifies the element–wise differences between the two matrices:

$$\text{Correlation Penalty} = \|C_{\text{real}} - C_{\text{synthetic}}\|_F \tag{4.15}$$

where $\|\cdot\|_F$ represents the Frobenius norm. The generator's objective function is modified to include this penalty term, ensuring that the synthetic data preserve the correlation structure of the original dataset. The overall loss function is formulated as follows.

$$\mathcal{L}_{\text{total}} = \lambda_{\text{adv}} \mathcal{L}_{\text{adv}} + \lambda_{\text{corr}} \cdot \text{Correlation Penalty} \tag{4.16}$$

Here, $\mathcal{L}_{\text{adv}}$ denotes the adversarial loss of the GAN, while λ_{adv} and λ_{corr} are hyperparameters that balance adversarial training with correlation preservation. By penalizing deviations in correlation structure, this approach encourages the synthetic data to maintain the intervariable dependencies and statistical properties observed in the original dataset.

4.6.3 Bayesian Network GAN (BNGAN)

The primary goal of this approach is to capture the attribute dependencies effectively and integrate them as auxiliary information for GANs. To achieve

this, a Bayesian Network (BN) is utilized to model the relationships between variables. BNs are well–suited for this task as they explicitly encode conditional dependencies, offering a structured and interpretable representation of variable interactions. The extracted dependencies are subsequently leveraged within a Conditional GAN (CGAN) [Mir14], providing additional guidance for generating realistic synthetic data. A CGAN extends the conventional GAN framework by incorporating auxiliary information into both the generator and discriminator. Unlike standard GANs, which generate data without additional constraints, CGANs condition the generation process on specific input data. This conditioning ensures that the generated samples adhere to predefined structural relationships, resulting in more coherent and meaningful synthetic data. The proposed methodology is outlined in Algorithm 11, followed by a detailed step–by–step explanation.

To implement this approach, dependencies among variables are first learned using a BN, as described in Section 4.5.1. The identified parent–child relationships are then encoded into dense vector representations (embeddings), which serve as guidance for the GAN. For each parent variable, a corresponding embedding layer is initialized, where the embedding dimension depends on the number of distinct categories in that variable. These layers are trained to map categorical values to continuous vector spaces, allowing semantic relationships to be captured based on the BN structure. The embeddings of the parent variables are concatenated to form a conditioning vector, encapsulating the combined influence of multiple parent attributes. This vector is further processed using dense layers to generate a refined representation, which serves as input to the CGAN, guiding its data generation process.

Embedding layers have been widely used for learning continuous vector representations of categorical variables [Mik+13], as seen in models like Word2Vec for text processing. These embeddings encode semantic similarities by mapping categorical values to a continuous space, where proximity between vectors reflects underlying relationships. A similar strategy is adopted here to capture probabilistic dependencies in BNs. By integrating these embeddings into a CGAN, the generated synthetic data preserves the structural relationships inherent in the original dataset while benefiting from the flexibility of the CGAN architecture.

4.6.4 Enforcing DP for the enhanced GAN synthesizers

Standard GANs lack inherent differential privacy guarantees as they do not explicitly limit the influence of any individual data point. To enforce DP, a common strategy is to apply Differentially Private Stochastic Gradient Descent (DPSGD) to the discriminator, as it directly interacts with the real data to distinguish between real and synthetic samples. Since the generator never accesses the original dataset, privacy–preserving mechanisms are not required at this stage.

For PCGAN and CSGAN, we assume that the auxiliary information used

Algorithm 11 Bayesian Network GAN

Require: $\mathbf{X_{train}}$, $\mathbf{y_{train}}$
Ensure: $\mathbf{D_{syn}}$, Performance Metrics
Step 1: Learning Variable Dependencies
Define: $df \leftarrow$ Dataset containing both $\mathbf{X_{train}}$ and $\mathbf{y_{train}}$
Initialize: $G \leftarrow \emptyset$
for each chunk in df **do**
    while no improvement in score **do**
        $G \leftarrow \text{modify}(G)$
        $\text{score}(G) \leftarrow \text{BIC}(G)$
    end while
    $G_{\text{chunk}} \leftarrow G$
end for
$G_{final} \leftarrow \bigcup G_{\text{chunk}}$
Step 2: Encoding Dependencies
for each **parent** $\in G_{\text{final}}$ **do**
    $\mathbf{e}_{\text{parent}} \leftarrow \text{Embedding}(\textbf{parent})$
end for
for each **child** $\in G_{\text{final}}$ **do**
    $\mathbf{e}_{\text{child}} \leftarrow \Big\|\mathbf{e}_{\text{parent}}$ ▷ Concatenate embeddings of parents
end for
Step 3: Define CGAN
$\mathcal{G} \leftarrow \text{Generator}(\mathbf{z}, \mathbf{e})$
$\mathcal{D} \leftarrow \text{Discriminator}(\mathbf{x}, \mathbf{e})$
$\mathcal{L}_{\text{adversarial}} \leftarrow \mathbb{E}[\log \mathcal{D}(\mathcal{G}(\mathbf{z}, \mathbf{e}))]$
$\mathcal{L}_{\text{reconstruction}} \leftarrow \mathbb{E}[\|\mathbf{x} - \mathcal{G}(\mathbf{z}, \mathbf{e})\|^2]$
Step 4: Training CGAN
for $t = 1$ **to** T **do** $\mathbf{x}_{real} \sim \mathbf{X_{train}}$
$\mathbf{x}_{syn} \leftarrow \mathcal{G}(\mathbf{z}, \mathbf{e})$
Train Discriminator: $\mathcal{L}_D \leftarrow \mathbb{E}[\log \mathcal{D}(\mathbf{x}_{real}, \mathbf{e})] + \mathbb{E}[\log(1 - \mathcal{D}(\mathbf{x}_{syn}, \mathbf{e}))]$
Train Generator: $\mathcal{L}_G \leftarrow \mathcal{L}_{\text{adversarial}}$
end for

is public knowledge. As a result, no additional privacy–preserving techniques are necessary beyond applying DPSGD to the discriminator. In the case of BNGAN, which utilizes a Bayesian network to model attribute dependencies and generates an embedding layer as input to the CGAN, we introduce an additional privacy mechanism by adding Laplace noise to the embeddings. Given that the values of the embeddings lie within the range of $[-1, 1]$, the maximum possible change between two neighboring datasets is at most 2. This value serves as the global sensitivity for the Laplace mechanism, ensuring that the added noise appropriately preserves DP. Applying DP at the embedding level provides a more fine–grained privacy guarantee that is difficult to achieve using k-anonymity. By combining DPSGD for the discriminator and Laplace noise for Bayesian network embeddings, BNGAN ensures robust privacy protection while maintaining the fidelity of the synthetic data. Applying k-anonymity before training the GAN distorts attribute relationships, disrupts dependency structures, and reduces data utility, leading to poor–quality embeddings and synthetic data. In contrast, DP preserves fine–grained patterns by adding controlled noise during training, ensuring both privacy and high–fidelity data generation.

4.7 Empirical Results

We now describe the architecture of the GAN, and the effects of synthetic data generated using the proposed techniques on ML performance, correlation similarity, and the effectiveness of differentially private synthetic data.

4.7.1 Conditional GAN (CGAN) Architecture

The Conditional GAN employed in this analysis consists of two primary components: the generator and the discriminator. The generator receives a concatenated noise vector and a conditioning vector as inputs. These are passed through four fully connected layers, each utilizing LeakyReLU activation, batch normalization for stable training, and dropout for regularization. The output is generated through a final dense layer. On the other hand, the discriminator takes both a data sample and the conditioning vector, concatenates them, and processes them through four fully connected layers with LeakyReLU and dropout. The output is classified as either real or synthetic using a sigmoid activation function in the final layer. Both the generator and the discriminator are trained using binary cross–entropy loss and the Adam optimizer. Unlike traditional GANs, this CGAN integrates a conditioning vector along with random noise, enabling the generation of domain–specific data that preserves the inherent attribute relationships, which is crucial for structured discrete datasets.

4.7.2 Impact of Synthetic Data on ML Performance

We assess the utility of synthetic data generated by four methods: CTGAN, PCGAN, CSGAN, and BNGAN, using multiple machine learning models. CTGAN serves as the baseline for comparison, as it is previously used and widely regarded as one of the most effective GAN architectures for synthesizing tabular data. For classification tasks on the Adult and German Credit Risk datasets, we use LightGBM, XGBoost, and Logistic Regression models, evaluating the performance based on accuracy. For the SD2011 dataset, income prediction is performed using LightGBM regression, XGBoost regression, and Linear Regression models, with performance measured by Root Mean Squared Error (RMSE). This comprehensive evaluation provides a thorough analysis of the utility of synthetic data across various tasks and datasets, as summarized in Table 4.5.

Each model is trained on synthetic data and evaluated on real, out–of–sample data. For the Adult dataset, BNGAN outperforms the other models with the highest accuracy, ranging from 0.78 to 0.79 , across all machine learning models. Similarly, for the SD2011 dataset, BNGAN demonstrates the lowest RMSE across all models, ranging from 0.42 to 0.45, with PCGAN achieving comparable results ranging from 0.43 to 0.46. In contrast, CTGAN shows much higher RMSE values 1185 to 1237 for the SD2011 dataset. We think that this is because it contains missing values and outliers, as no pre–processing was applied. This highlights the challenges of working with raw, unprocessed data and emphasizes the importance of incorporating structure into synthetic data generation.

For the German Credit Risk dataset, BNGAN again achieves the highest accuracy across all models ranging from 0.68 to 0.74, demonstrating the benefits of using a Bayesian network to capture dependencies between attributes. By modeling these dependencies, BNGAN generates more realistic synthetic data, leading to improved performance in machine learning models. On the other hand, CSGAN consistently yields the lowest utility among all methods for both classification tasks. Additionally, we compared the performance of synthetic data against the original data. As anticipated, we observed a slight decline in machine learning performance with synthetic data, which aligns with the goal of synthetic data: to approximate the original distribution rather than surpass it.

For the Adult dataset, we compared our results with the previously proposed M–KCTGAN approach. As shown earlier in Table 4.2, the $F1$ score achieved by M–KCTGAN was 0.591. In contrast, our proposed BNGAN model achieved a higher $F1$ score of 0.66. Given that the Adult dataset is the most imbalanced among those considered, reporting the $F1$ score is particularly relevant, as it better reflects the model's performance on minority classes. This improvement demonstrates that incorporating prior knowledge into the generative process effectively helps address class imbalance.

Table 4.5: Utility evaluations for ML models trained on synthetic data and tested on real out–of–sample data

Data	Utility Metric	ML Model	Synthetic Data				Original
			CTGAN	PCGAN	CSGAN	BNGAN	Data
ADULT	Accuracy ↑	LightGBM	0.75	0.74	0.70	**0.79**	0.87
		XGBoostC	0.75	0.73	0.69	**0.79**	0.86
		LogisticR	0.74	0.74	0.71	0.78	0.86
Credit Risk	Accuracy ↑	LightGBM	0.66	0.61	0.58	**0.74**	0.75
		XGBoostC	0.65	0.62	0.56	0.68	0.76
		LogisticR	0.67	0.63	0.59	0.70	0.74
SD2011	RMSE ↓	LightGBM	1207.35	0.44	0.48	0.43	1050.31
		XGBoostR	1236.80	0.46	0.50	0.45	1091.21
		LinearR	1185.21	0.43	0.47	**0.42**	1015.82

4.7.3 Impact of Synthetic Data on Attribute Correlations

Preserving the pairwise correlations between the attributes in synthetic data is an important aspect of data utility. To assess this, we used Cramér's V with bias correction [Ber13], a commonly adopted measure for evaluating the strength of relationships between pairs of categorical attributes, as detailed in the literature [Tao+21]. Cramér's V is defined as:

$$V = \sqrt{\frac{\chi^2}{n \cdot \min(k-1, r-1)}} \tag{4.17}$$

where χ^2 is the chi–squared statistic, n is the total number of observations, k is the number of categories in the first variable, and r is the number of categories in the second variable. The Cramér's V values are grouped into four categories: low (V $\in$ [0, 0.1)), weak (V $\in$ [0.1, 0.3)), middle (V $\in$ [0.3, 0.5)), and strong (V $\in$ [0.5, 1)). To evaluate how well the synthetic data replicates the original data, we use a correlation accuracy metric for categorical attributes. This metric calculates the percentage of attribute pairs in the synthetic data that exhibit the same correlation levels as in the original data.

The results, shown in Table 4.6, reveal that different methods of generating synthetic data vary in their ability to preserve attribute relationships. In the case of the Adult dataset, which has a high class imbalance, correlation accuracy is relatively low across all methods. This is likely due to the minority class attributes not being well represented in the synthetic data, resulting in weaker correlations between attributes. For the SD2011 dataset, PCGAN achieved the highest correlation accuracy of 0.6915, likely due to its effective enforcement of domain–specific constraints in the loss function, which helps the model better capture relationships between attributes. BNGAN also performed well, achieving a correlation accuracy of 0.6780, reflecting the advantages of incorporating a Bayesian Network model to capture dependencies between attributes and effectively preserve correlations. Similar trends were observed with the

Table 4.6: Correlation Accuracy and Similarity for Categorical and Numerical Attributes

	Categorical				Numerical			
Dataset	CTGAN	PCGAN	CSGAN	BNGAN	CTGAN	PCGAN	CSGAN	BNGAN
ADULT ↑	0.3626	**0.4190**	0.3524	0.3714	0.8581	0.8843	0.8718	**0.8932**
Credit Risk ↑	0.6723	0.6812	0.6235	**0.6981**	0.8642	**0.8714**	0.8312	0.8711
SD2011 ↑	0.6684	**0.6915**	0.6123	0.6780	0.9758	0.9916	0.9468	**0.9971**

Credit Risk dataset, where BNGAN achieved the highest correlation accuracy of 0.6981, slightly outperforming PCGAN. Once again, CSGAN exhibited the weakest performance in all settings.

In addition to categorical correlations, we also evaluate the preservation of relationships between numerical attributes by computing the Pearson correlation coefficient [Coh+09] for both real and synthetic data. We compute two correlation values: $R_{A,B}$ for the real data and $S_{A,B}$ for the synthetic data. The similarity between these values is quantified using the following formula:

$$\text{score} = 1 - \frac{|S_{A,B} - R_{A,B}|}{2} \tag{4.18}$$

A score of 1 indicates perfect similarity, while a score of 0 indicates no similarity. This method, adapted from SD Metrics [Dat23], provides a standardized way to evaluate data quality. The results show that BNGAN consistently achieved the highest correlation similarity scores, particularly for the SD2011 and Adult datasets, indicating its effectiveness in preserving numerical relationships. PCGAN also performed well, particularly on the SD2011 and Credit Risk datasets, by enforcing constraints in the loss function. In contrast, CSGAN, which relies on correlation–based penalties, achieved lower correlation similarity scores, suggesting that it may struggle to fully capture the complex dependencies between attributes. Overall, constraint–based approaches such as PCGAN and BNGAN outperform CSGAN, with BNGAN demonstrating the strongest ability to preserve both categorical and numerical correlations across multiple datasets.

4.7.4 Impact of Differentially Private Synthetic Data

To ensure that the synthetic data generation process adheres to Differential Privacy, we implemented a DP mechanism in our proposed GAN models, as outlined in Section 4.6.4. We assessed the efficacy of these models by comparing them with three baseline methods: DPGAN, PATEGAN, and ADSGAN. Table 4.7 presents the machine learning performance when models are trained with DP, where we set $\epsilon = 1$ and $\delta = \frac{1}{N}$.

For our DP-BNGAN model, we apply noise injection at two stages: first, during the generation of Bayesian network–based embeddings with $\epsilon = 1$, and

Table 4.7: ML performance using Differential Privacy

Dataset	Utility Metric	DP-PCGAN	DP-CSGAN	DP-BNGAN	DPGAN	PATEGAN	ADSGAN
ADULT	Accuracy ↑	0.65	0.67	**0.72**	0.54	0.69	0.71
Credit Risk	Accuracy ↑	0.62	0.40	0.66	0.54	**0.96**	0.82
SD2011	RMSE ↓	**0.48**	0.57	0.51	0.61	0.58	0.49

second in the discriminator component of the CGAN, also with $\epsilon = 1$. Consequently, the total privacy budget for DP-BNGAN is $\epsilon = 2$. We assessed model utility accross three datasets. For the Adult and Credit Risk datasets, classification accuracy was measured using LightGBM, the highest–performing model from Table 4.5. For the SD2011 dataset, we evaluated prediction performance using RMSE with linear regression, which was also the top performer in Table 4.5.

The results reveal that each model performed differently across datasets, highlighting their ability to adapt to the specific characteristics of each dataset while maintaining privacy. For the Adult dataset, DP–BNGAN yielded the best performance, demonstrating its ability to capture complex data distributions while ensuring privacy. PATEGAN performed best on the Credit Risk dataset, likely due to its enhanced learning capabilities. In contrast, DPGAN showed the lowest performance across all datasets. Although the inclusion of DP slightly reduced model performance, the results indicate that the models still maintained utility comparable to the baseline methods, suggesting that a balance between privacy and utility can be achieved. Moreover, there is potential to further improve utility at the cost of a reduced privacy guarantee.

4.7.5 Discussion

So far, we have explored synthetic data generators from multiple perspectives. We began by focusing on generating high–quality synthetic data for real–world high–dimensional datasets, emphasizing the role of manifold learning and the need for privacy protection. While some approaches yielded promising results, others exhibited suboptimal performance, highlighting the challenges in preserving both data utility and privacy. To enhance performance, we then investigated whether attribute dependencies could be effectively preserved and explored strategies to incorporate prior knowledge into GANs. Our goal was to determine whether structural dependencies and domain constraints could improve the quality and realism of synthetic data.

However, despite these advancements, GANs and VAEs remain largely black–box in nature, raising concerns about their distributional capabilities, whether they truly capture the underlying data distribution or merely learn superficial patterns. Understanding these distributional properties is critical to ensure that synthetic data faithfully represents real–world distributions rather than producing unrealistic or biased samples. Thus, we now shift our focus to systematically analyze the distributional characteristics of these models, ad-

dressing their interpretability and reliability in real–world applications.

4.8 Explore Distribution Learning of Synthetic Data Generators

Until now, we have explored various synthetic data generation techniques. While these methods are highly effective in generating synthetic data, their black–box nature makes it challenging to interpret and analyze their learning behavior, especially with complex datasets [Den+09]. To address this, we utilize simpler, artificially constructed datasets such as Swiss Roll and S–Curve, which can be visualized in lower–dimensional spaces. These datasets provide clearer insights into the learning dynamics of generative models and their ability to capture data distributions. Although GANs have demonstrated strong performance in certain applications [Kar+17], their effectiveness can vary when handling complex data distributions [BDS18; OOS17]. By focusing on artificially generated datasets, we aim to analyze the learnability of fundamental data structures in low–dimensional spaces. This approach enables a more transparent evaluation of synthetic data generation models, helping us to uncover their strengths, limitations, and broader applicability across different domains. We adopted the following framework to visualize that the manifolds generated using synthetic data generators converge to real data manifolds.

1. **Dataset Selection**: We start by selecting a real–world high–dimensional dataset exhibiting the manifold structure such that the dataset is in the topological space which is not Euclidean. We have used MNIST [Den+09] dataset for this task.

2. **Train a Manifold Learning Model**: Utilize Uniform Manifold Approximation and Projection (UMAP) [MHM18], to train a model on the data set chosen from the previous step. UMAP is specifically selected for its ability to preserve both local and global structures within the dataset, also because of its inverse transformation function. This characteristic makes UMAP more scalable compared to other manifold learning techniques. Furthermore, UMAP boasts faster computation times than t–SNE, which enhances its practicality. Through this step, the high–dimensional dataset is transformed into a lower–dimensional Euclidean space, facilitating visualization. Visualization becomes feasible in this latent space, enabling a better understanding of the data's intrinsic structure.

3. **Reconstruction to Original Space**: Employ the inverse mapping function of UMAP, to reconstruct the transformed data from the previous step back to its original high–dimensional space using the same dataset chosen from step 1. This process ensures that the model is trained to han-

dle data sets effectively with manifold structures, leveraging the insights gained from high–dimensional data sets in the real world.

4. **Generation of Artificial Data**: Now we generate artificial datasets created in $\mathbb{R}^4$ and $\mathbb{R}^2$ such as S-Curve and Swish roll dataset to visualize and understand the dataset, and their lower–dimensional transformations, which is not feasible with high–dimensional real–world data.

5. **Test the Manifold Learning Model**: Apply the trained manifold learning model from Step 2 to transform artificial datasets into the latent space for visualization. This step assesses the model's ability to preserve proximity between points from high–dimensional to lower–dimensional spaces and allows for performance evaluation of the manifold learning algorithm.

6. **Synthetic Data Generation**: This step involves using models such as GANs and VAEs to generate synthetic data from a learned latent space. The goal is to assess how effectively the generated data captures the statistical properties of real data while introducing sufficient variability to prevent direct replication. This approach enhances privacy by maintaining the underlying data structure without exposing exact original records.

7. **Synthetic Data Reconstruction**: Apply the inverse transformation function of UMAP model to map the generated synthetic data back to its original high–dimensional space. Assess the fidelity of this reconstruction by comparing it with the original synthetic data, ensuring that key structural and statistical properties are preserved. This evaluation helps to determine how well the transformation process retains the integrity of the data.

We carefully selected a range of datasets with manifold structures to comprehensively evaluate our approach. Manifold structure represents intricate data distribution patterns in a high–dimensional space that can't be fully captured by traditional linear methods. The selected datasets offer different complexities and characteristics, allowing us to assess the performance of generators across diverse scenarios. These datasets include point datasets like the Swish Roll dataset [Mar11], S–curve dataset [Ped+11], Concentric circles [Ped+11], Mixture of Gaussian points [Ped+11], and Two–Half Circles datasets [Ped+11]. The first two datasets reside in $\mathbb{R}^4$, with points distributed across a 3D plane and their labels in another dimension. The remaining datasets are in the $\mathbb{R}^2$ plane. Each dataset comprises 4000 samples, with each sample representing a fixed point in $\mathbb{R}^n$. All datasets were generated using the sklearn library [Ped+11]. For instance, the Swish Roll and S–curve datasets provide examples of non–linear structures in higher–dimensional spaces, while datasets like Concentric circles and Mixture of Gaussian points showcase different patterns in two–dimensional spaces with some degree of discontinuity. By including

datasets with varying complexities, we aim to comprehensively test the robustness and effectiveness of our manifold learning techniques and synthetic data generators. Additionally, the MNIST dataset, with its images of handwritten digits, offers a real–world example where manifold learning can be applied to understand and generate complex data distributions.

We implemented multiple generative models for synthetic data generation, starting with a Vanilla GAN. The generator comprised four dense layers with Leaky ReLU activations and a tanh output, while the discriminator mirrored this structure with a sigmoid activation. After updating the discriminator, it was frozen, and the generator was trained on fake data, with loss back propagated to adjust its weights. For improved image synthesis, we employed a Deep Convolutional GAN (DCGAN), integrating Batch Normalization and a Convolution1D layer for upsampling. Unlike Vanilla GAN, DCGAN's discriminator processed image–like inputs instead of vectors, using Leaky ReLU activations for stability.

To handle tabular data, we used Conditional Tabular GAN (CTGAN), while Differentially Private GAN (DPGAN) was implemented to introduce privacy guarantees by injecting controlled noise during training. Additionally, we explored Variational Autoencoders (VAEs), with an encoder learning the latent distribution's mean and log–variance, and a decoder reconstructing samples via a sigmoid output. The reparameterization trick ensured smooth gradient flow through the stochastic layer. All models—Vanilla GAN, DCGAN, CTGAN, DPGAN, and VAE—were trained using the Adam optimizer with a learning rate of 10^{-4} ensuring stable convergence and high–quality synthetic data generation.

We now visualize the distributional capabilities of synthetic data generators on artificially created datasets.

4.8.1 Visualize Synthetic Generation with S-Curve Dataset

Figure 4.3 illustrates the step-by-step application of our methodology to the S-Curve dataset. We began with the original S-Curve dataset, generated using the *make-s-curve* function from the sklearn library. Then, we applied the pre–trained UMAP manifold learning model, originally trained on the MNIST dataset, to project the S-Curve data into a 2D plane (Figure 4.3b). This transformation effectively preserved the data's shape, curves, and geometry, with the univariate positioning of the samples (highlighted by colored labels) maintaining the original structure. This validates the manifold hypothesis, indicating that points close in high–dimensional space remain close in the lower–dimensional representation.

Next, we generated synthetic data points using a VAE, based on the manifold–transformed data (Figure 4.3c). The VAE, a generative model that learns the underlying distribution of input data by encoding it into a lower–dimensional latent space and decoding it back, was able to leverage the manifold–transformed data. This allowed the VAE to generate synthetic data that aligns with the

original dataset's patterns and distribution. The S-curve structure observed in the latent space further confirms the VAE's ability to capture essential features and variations of the data.

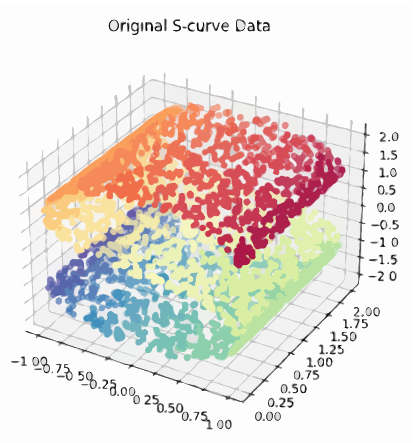
Finally, we used the inverse transform function of the manifold model to reconstruct the data in the original space (Figure 4.3d). While the reconstructed data points clustered around the central region, the dispersion was limited, indicating that the inverse transform struggled to fully recover the high–dimensional structure. This suggests potential improvements, such as refining the transformation process to better preserve the details of the original data distribution.

4.8.2 Unrolling the Swish Roll:Manifold Transformation

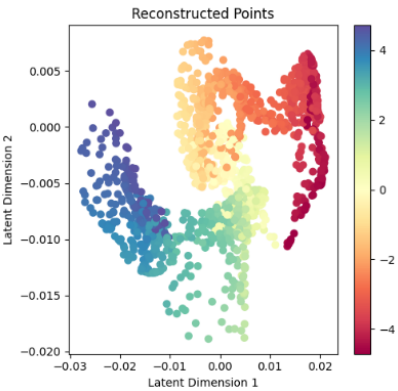
Figure 4.4 presents the Swish roll dataset, first depicted in Figure 4.4a. When transformed into a two–dimensional space using UMAP (Figure 4.4b), the dataset's underlying structure and relationships become more apparent, demonstrating the principles of manifold learning. UMAP efficiently maps complex high–dimensional patterns into a lower–dimensional space while preserving key characteristics, making it valuable for meaningful data representation. In Figure 4.4c, synthetic data generated by a VAE from the UMAP–transformed points retains distinct labels and closely resembles the original dataset. Minor variations introduce controlled noise, enhancing privacy while maintaining key data properties. Finally, Figure 4.4d illustrates the reconstruction of the original dataset using the inverse transform of the manifold model. While labeled points tend to cluster, overlapping regions present challenges for precise reconstruction, underscoring the difficulties of managing high–dimensional data with noise and intersecting surfaces.

4.8.3 Understanding 2D Point Datasets

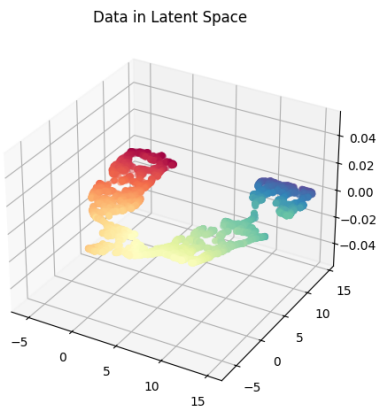
In our evaluation of synthetic data generators, we applied our methodology to 2D point datasets. Figure 4.5 illustrates Gaussian clusters generated using the make-blobs function from sklearn, where three clusters are defined with a standard deviation of 0.2. To assess the generative capabilities of a VAE, we trained the model on this dataset and visualized the generated data. The synthetic points form three distinct clusters, demonstrating the VAE's ability to approximate the original distribution. However, some intra–cluster stretching is observed, reflecting minor deviations that contribute to privacy preservation. Despite these distortions, the VAE effectively captures the discrete nature of the dataset, preserving the underlying cluster structure. This outcome contrasts with the findings in [RM19], where VAEs exhibited challenges in handling discontinuous data due to their assumption of a continuous latent space. However, our results suggest that while the VAE introduces minor distortions, it successfully models discontinuous data distributions without collapsing the distinct clusters.



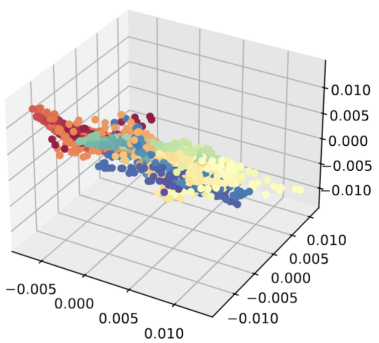
(a) Original Data



(b) Data Transform



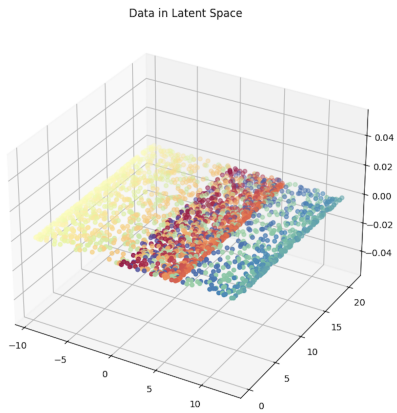
(c) Synthetic Data



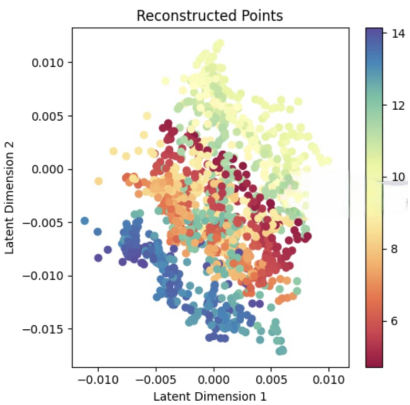
(d) Reconstructed data

Figure 4.3: S-Curve Dataset

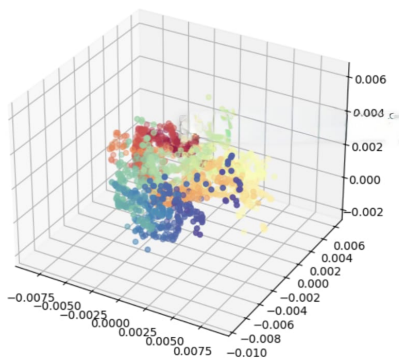
(a) Original Data



(b) Data Transform



(c) Synthetic Data



(d) Reconstructed Data

Figure 4.4: Swish Roll Dataset

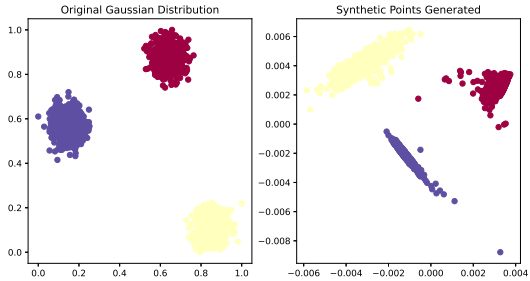


Figure 4.5: Mix of Gaussian Points

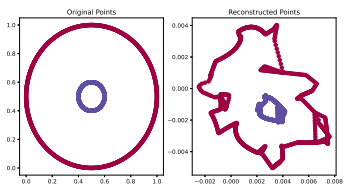


Figure 4.6: Concentric Circles

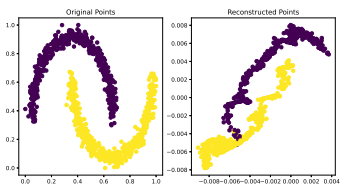
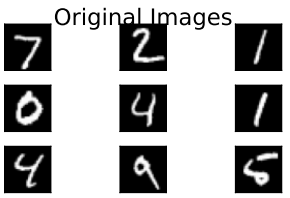


Figure 4.7: Two Half Circles

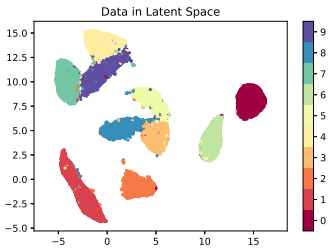
A similar pattern is observed in Figures 4.6 and 4.7, which presents results for the concentric circles and two half-circles datasets. Both original datasets exhibit inherent discontinuities with well–defined geometric structures. The VAE generated synthetic data closely follow these patterns while incorporating slight variations. This behavior arises from the VAE learning the probability distribution of the input data and sampling new points accordingly. The minor deviations introduced by the VAE ensure that key structural and geometric properties of the original dataset are preserved while enhancing privacy through controlled noise.

4.8.4 Visualizing Real–World Dataset

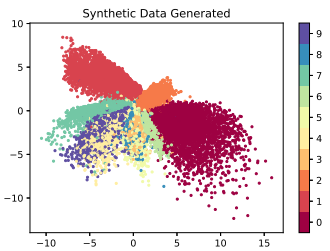
We investigated the effectiveness of manifold learning and synthetic data generation using the MNIST dataset, a widely used benchmark in machine learning. The original MNIST dataset consists of 70,000 handwritten digit images, each sized at 28×28 pixels. Figure 4.8a presents the raw dataset before any transformations. To explore its structure, we applied manifold learning to project the high–dimensional image data into a 2D latent space, as shown in Figure 4.8b. This transformation results in well–separated clusters, where each cluster corresponds to a distinct digit class. Using this lower–dimensional representation, we then trained a VAE to generate synthetic data. The resulting synthetic samples, visualized in Figure 4.8c, exhibit tight clustering, indicating



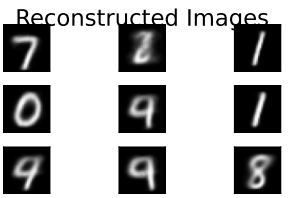
(a) Original Data



(b) Data transformed



(c) Synthetic Data



(d) Reconstructed Data

Figure 4.8: MNIST Dataset

that the VAE effectively captures the underlying data distribution. Finally, Figure 4.8d shows the reconstructed data, which closely resembles the original images, demonstrating the VAE's ability to learn and regenerate realistic samples. We achieved similar results with GAN on MNIST dataset.

However, when working with real–world image datasets like MNIST, visualization is inherently limited. While we can observe and compare the original and reconstructed images, once the data is projected into a latent space, direct visualization is restricted to examining the spatial distribution of data points based on their labels. This black–box nature of synthetic data generators makes it challenging to interpret how the data's geometry evolves through each transformation stage. To address this limitation, we incorporated artificially created datasets in 4D and 2D spaces, enabling a more transparent visualization of how synthetic data generators learn and preserve the structural properties of the data.

4.8.5 Privacy Risk Assessment in VAE

We extended our privacy analysis of VAE by introducing artificial points into the original S-Curve dataset and evaluating the model's ability to regenerate them. Figure 4.9a presents the modified dataset, where 10% additional points are systematically placed along a straight line and highlighted in red

and green. The corresponding synthetic data, generated by the VAE, is shown in Figure 4.9b, where the newly introduced points are successfully regenerated. This indicates that the VAE effectively learns and generalizes from the dataset while preserving its overall structure, including the added data points. When newly introduced points are numerous and systematically distributed, as in Figure 4.9a, the VAE's ability to regenerate them in Figure 4.9b suggests that it has effectively captured the dataset's underlying distribution. This reinforces the VAE's capability to generate realistic synthetic data while maintaining structural consistency with the original dataset.

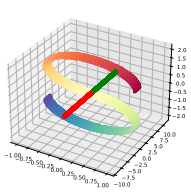
However, if the VAE reproduces only a small fraction of the added points in their exact original locations, this may indicate a privacy risk due to memorization rather than learning general patterns. Figure 4.9c and 4.9d illustrate this scenario. In Figure 4.9c, only 0.01% of the dataset consists of newly added points, strategically placed within the S-Curve. Figure 4.9d presents the VAE–generated synthetic data, where the newly added points appear in different positions rather than being replicated exactly. This suggests that the model has learned general patterns instead of memorizing specific samples.

A closer examination of Figure 4.9c reveals that three newly introduced points (two green and one red) coincide with the S-Curve structure. However, in Figure 4.9d, these points are not regenerated at their original locations but are instead distributed throughout the S-Curve, further indicating that the VAE has not memorized the exact data points. The scattered placement of these points confirms that the VAE preserves the overall S-Curve structure while preventing direct replication of individual samples, thereby reducing the risk of privacy leakage.

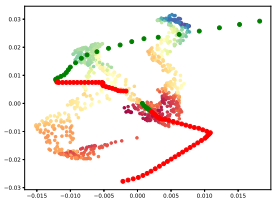
This analysis highlights the importance of assessing privacy risks in synthetic data generation. If a VAE precisely regenerates specific added points, it suggests memorization of individual data samples, potentially compromising privacy. To mitigate this risk, it is crucial to ensure that the VAE captures the general distribution of the data without retaining identifiable information. This prevents potential privacy breaches and safeguards sensitive data while enabling realistic synthetic data generation.

4.8.6 Visualization with Diverse GAN Architectures

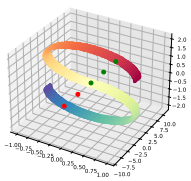
We evaluated multiple GANs for synthetic data generation, including Vanilla GAN, Deep Convolutional GAN (DCGAN), Conditional Tabular GAN (CTGAN), and Differentially Private GAN (DPGAN). Our assessment began with Vanilla GAN applied to the S-Curve dataset, as shown in Figure 4.10a. The generated data exhibited poor alignment with the original points, indicating that Vanilla GAN struggled to capture the intrinsic 3D structure of the dataset. A similar trend was observed across other datasets, but for clarity, we present only the results for the S-Curve dataset. Next, we applied DCGAN to the Swish Roll dataset in Figure 4.10b. While the generated points overlapped with the original data, their distribution appeared disorganized, suggesting



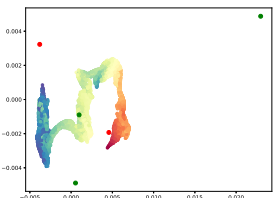
(a) Data with 10% new points



(b) Regenerated points



(c) Data with 0.01% new points

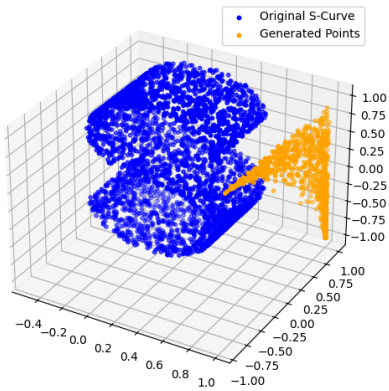


(d) Regenerated points

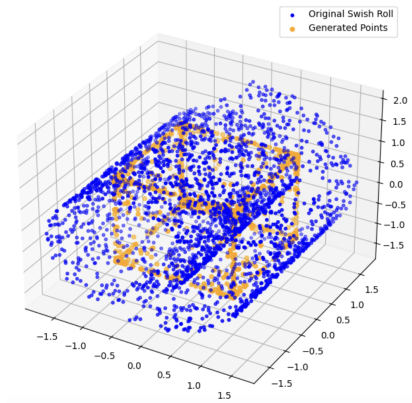
Figure 4.9: Privacy Risk Assessment in VAE

that DCGAN had difficulty preserving the dataset's geometric structure. Similarly, when using CTGAN on the S-Curve dataset in Figure 4.10c, we observed significant overlap between synthetic and real data points, but the latent space representation remained scattered. This suggests that CTGAN struggled to learn the 3D geometry effectively. Finally, we tested DPGAN on the Swish Roll dataset in Figure 4.10d. However, the generated points were dispersed without preserving the dataset's original geometric properties, demonstrating DPGAN's difficulty in maintaining structural fidelity. GANs, in general, are known for instability and slow convergence [Goo+20; RMC15], particularly when applied to manifold data. In contrast, VAEs provided more consistent and reliable results. Unlike GANs, which often suffers from mode collapse, VAEs effectively captured the diverse structures present in our datasets [HYW18].

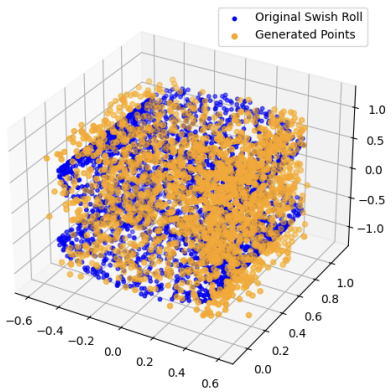
This discrepancy highlights a key challenge: while GANs excel in generating complex, structured data like images, they often struggle with simple, low–dimensional point cloud datasets such as Swiss Roll or S-Curve. This is primarily due to the nature of their latent space, which is typically modeled as a dense and unstructured distribution in $\mathbb{R}^n$, making it difficult to capture the intrinsic geometry of manifolds with sparse or highly non–linear structures. In contrast, VAEs tend to offer greater stability and better manifold alignment, making them more effective for synthetic data generation in such settings. However, for real–world datasets like MNIST, which have more structured and well–sampled distributions, both GANs and VAEs perform comparably well.



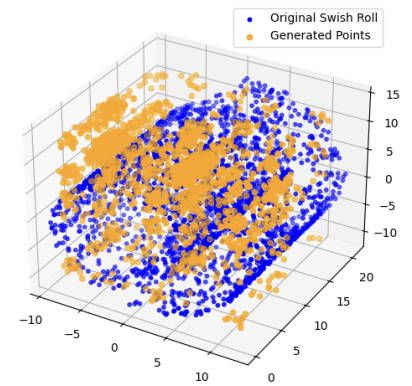
(a) VanillaGAN



(b) DCGAN



(c) CTGAN



(d) DPGAN

Figure 4.10: Results with other types of GAN

4.9 Conclusion

In this chapter, we explored the generation of high–quality synthetic data as an alternative to traditional anonymization, particularly for high–dimensional datasets where generalization or suppression may lead to excessive information loss. To preserve the intrinsic structure of data, we leveraged manifold learning and integrated k–anonymity to protect training data before synthetic data generation. We evaluated the generated data using statistical and ML utility assessments and conducted data reconstruction attacks to analyze privacy vulnerabilities. While our approach outperformed existing baselines, some suboptimal cases emphasized the need to better preserve data correlations for improved generation quality.

To address these limitations, we explored the role of prior knowledge in GANs, incorporating public constraints, correlation structures, and Bayesian networks to improve data realism. Given that GANs are susceptible to adversarial attacks, we introduced differential privacy mechanisms (DPSGD, Laplace noise for Bayesian network embeddings) to enhance robustness. Our comparison across three tabular datasets demonstrated that prior knowledge improves both the quality and privacy of synthetic data.

Beyond evaluating synthetic data quality, we analyzed the black–box nature of generators by examining their distribution learning capabilities, latent space representations, and privacy risks. Our findings revealed that VAEs outperform GANs for simple, low–dimensional datasets due to their structured latent space and stable training, whereas GANs struggle with mode collapse and instability. However, for complex datasets such as MNIST, both models perform comparably, as GANs leverage adversarial training to capture richer data distributions. Additionally, VAEs inherently offer stronger privacy protection, while GANs require careful tuning to mitigate privacy risks. Overall, our study underscores the challenges of understanding and improving synthetic data generators. In the future, other generative models such as diffusion models could be explored to assess the quality of synthetic data, particularly in the context of high–dimensional and tabular datasets.

Chapter 5

Privacy-Aware Language Models

> We are getting to the point where the machines are going to surpass us, and it's going to happen much faster than anyone thinks
>
> — *Geoffrey Hinton*

So far, we have focused on protecting the sensitive information of individuals that resides in high–dimensional databases. However, high dimensionality is not only a concern for data but also for modern machine learning models, particularly foundation models, which consist of millions of parameters and exhibit high inference times. Many of these models are also vulnerable to memorizing the sensitive training data, which poses significant privacy risks. In this chapter, we study our third Research Question **RQ3** and explore methods to improve the efficiency and privacy of such large–scale models. Specifically, we focus on techniques to reduce the inference time while simultaneously mitigating privacy risks. A prime example of such models is language models, which have gained immense popularity due to their recent advancements but also present significant computational and privacy challenges. The objective of this chapter is to make these models more practical for real–world deployment by enhancing their efficiency and ensuring robust privacy protection.

5.1 Problem Formulation

Transformer–based models, particularly large–scale language models such as BERT [Dev18] and GPT [Bro20], have significantly advanced natural language understanding (NLU). Their growing size and complexity have enabled remarkable improvements in accuracy and emergent capabilities [Cho+23; Bro20].

However, their widespread deployment is hindered by two major challenges:

High Computational Overhead. Large Language Models (LLMs) contain millions to billions of parameters, making both training and inference time highly computational intensive. The time taken by a model to generate a response to a query is referred to as inference time. For real–time applications such as next–word prediction or sentence completion, achieving a response time within milliseconds is essential for usability. However, the large size of these models due to large training parameters significantly increases the inference time, creating a major bottleneck in their widespread adoption for real–world production scenarios [Xu+23]. To enable seamless integration into practical applications, it is crucial to develop techniques that reduce the number of model parameters, thereby decreasing the inference time, while preserving the model performance. Several approaches have been proposed to achieve this, including pruning [HP88], knowledge distillation [BCN06], and quantization [Wu+16; Gon+14]. These methods focus on compressing the model by reducing the number of parameters, thereby lowering the memory and computational requirements. A compressed model with fewer parameters results in faster inference times, making LLMs more suitable for real–time applications. It's commonly observed that deep learning models often contain many redundant parameters that could be removed while still maintaining performance [Naj+15]. Thus, addressing this challenge is pivotal to unlocking the full potential of transformer–based technologies and realizing their impact at scale.

Privacy Vulnerabilities. Safeguarding privacy emerges as another critical concern while deploying language models. The utilization of personal sensitive data for model training makes it susceptible to inadvertent disclosure, raising formidable privacy implications, as input text or its vector representations can inadvertently expose private information [CN18; LBC18]. Recent studies have demonstrated that language models can remember the training data, which can lead to privacy attacks [Car+19; Car+21]. In LLMs, techniques like prompt engineering can exploit this memorization, potentially extracting confidential or personally identifiable information from the model, which poses a critical risk when models are deployed in real–world applications. While there are several methods to protect privacy, automatic de–identification is a widely explored approach for removing personally identifiable information. Some authors [Vak+22; VD22] evaluated the effects of pre–training and fine–tuning BERT models on both de–identified and original datasets, employing techniques such as pseudo anonymization and sentence removal to protect sensitive content. However, to achieve stronger and more generalizable privacy guarantees, Differential privacy can be incorporated into the training process, providing formal protection against data leakage. While DP mitigates these risks by adding noise during training, language models still face privacy challenges, such as vulnerability to inference–time attacks [Sho+17], where sensitive information can be inferred from the model outputs and repeated queries can weaken the privacy guarantees. Achieving DP or model compression for the

LLM typically involves a trade–off with utility loss. Attempting both simultaneously can result in significant utility loss, highlighting the complexity of this challenge.

Some studies have investigated model compression in ensemble learning under privacy constraints, notably through the Private Aggregation of Teacher Ensembles (PATE) framework [Pap+16; Pap+18]. In PATE, an ensemble of teacher models is trained on disjoint subsets of sensitive data, and their aggregated outputs are used to supervise a student model on a separate, public or non–sensitive dataset. The aggregation mechanism, often combined with DP, ensures that individual data points in the teachers' training sets are protected during student model training. However, while PATE provides strong privacy guarantees for the training data of teacher models, the student model itself may still access private or auxiliary data during training, which introduces a potential vulnerability to privacy leakage. Moreover, if the student model is later fine–tuned or deployed in settings where it encounters sensitive data, the absence of built–in privacy protections during this phase can further compromise privacy. Additionally, several model compression frameworks have been explored in the context of differential privacy, including Differentially Private Knowledge Distillation (DPKD) and Differentially Private Iterative Magnitude Pruning (DPIMP), as discussed in [Mir+22]. In the DPKD approach, traditional knowledge distillation is applied, where both the teacher and student models are trained using DPSGD. The authors also emphasize the significance of random initialization for the student model to ensure effective learning under privacy constraints. In the DPIMP approach, the authors integrate magnitude–based pruning with DPSGD, resulting in a more efficient compressed model. Their results indicate that the pruning–based strategy outperforms the distillation–based approach in terms of model utility. However, in our work, we compare both approaches and demonstrate that our proposed method achieves superior performance, balancing both privacy guarantees and model utility more effectively.

Addressing both model compression techniques to reduce inference time and ensuring that these models remain privacy–preserving presents a significant challenge. While methods such as pruning, knowledge distillation, and quantization effectively reduce computational costs, they may inadvertently expose models to privacy risks, such as information leakage or membership inference attacks. Balancing efficiency and privacy remains an open problem, requiring novel techniques that integrate compression with robust privacy–preserving mechanisms. In this chapter, we explore strategies to achieve this balance, ensuring that the models maintain both low–latency performance and strong privacy guarantees, making LLMs practical, secure, and scalable in real–world applications.

We chose to work with BERT models in this study because they serve as the foundational models with a relatively smaller architecture, making them easier to interpret and analyze. While more recent models, such as LLaMA, offer advanced capabilities, their significantly larger size introduces additional

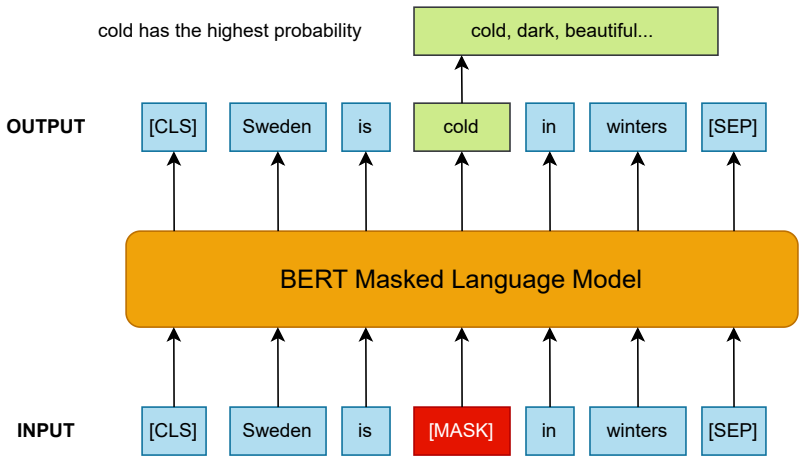


Figure 5.1: BERT Model

complexity, making it challenging to examine their internal mechanisms. However, since both BERT and LLaMA models are based on the transformer architecture and rely on attention mechanisms for language understanding, the proposed approach is expected to be applicable to LLaMA models as well. We describe the traditional BERT model as follows.

5.2 BERT Model

BERT (Bidirectional Encoder Representations from Transformers) [Dev18] is a deep learning model developed by Google that has revolutionized Natural Language Processing (NLP) by introducing a bidirectional approach to understanding the text. Unlike traditional language models that process text in a left–to–right or right–to–left manner, BERT reads entire sequences of words at once, capturing contextual relationships from both directions. This bidirectional training enables BERT to understand the meaning of words in context, improving tasks like question answering, text classification, and named entity recognition. BERT is pre–trained on massive corpora using a masked language model (MLM) objective, where random words in a sentence are masked and the model learns to predict them based on surrounding context, making it highly effective in transfer learning for various NLP tasks.

Figure 5.1 illustrates the core idea behind BERT's MLM objective. In this approach, a portion of the input tokens is randomly replaced with a special [MASK] token during pre–training. For example, in the sentence "Sweden is [MASK] in winters", the model is tasked with predicting the original word (*cold*) that was masked, using the context provided by the surrounding words.

As shown in the figure, the input tokens are first fed into the BERT model, which processes them using multiple transformer layers. The output layer then attempts to predict the most likely token to replace the masked word, leveraging bidirectional context. In this case, the model assigns the highest probability to "cold", among other possible tokens like "dark"or "beautiful". This masked language modeling enables BERT to learn deep contextual embeddings for words, allowing it to generalize effectively across downstream NLP tasks.

The architecture of BERT is based on the Transformer model, specifically utilizing multiple layers of self–attention and feedforward neural networks. It comes in different sizes, with BERT–base consisting of 12 transformer layers and 110 million parameters, while BERT–large has 24 layers and 340 million parameters. The model is fine–tuned on specific NLP tasks by adding task–specific layers on top of its pre–trained representations. During fine–tuning, BERT adjusts its weights to optimize for the given task, ensuring adaptability across diverse language applications. Due to its bidirectional nature and deep representation learning, BERT has set new benchmarks for NLP, significantly improving the performance of models in understanding complex linguistic structures.

5.3 Approach 1: Task–Specific Knowledge Distillation with DP

To reduce model inference time while ensuring strong privacy guarantees, we propose our first methodology titled *Task–Specific Knowledge Distillation with Differential Privacy*. We leverage Knowledge Distillation (KD) (as discussed in Chapter 2) as a model compression technique to transfer knowledge from a large teacher model to a more efficient student model while preserving data privacy through DP. KD is particularly suitable for our objective as it enables model compression, knowledge transfer, and improved generalization of the student model. While traditional KD has been extensively studied, it often overlooks task–specific features, motivating our task–specific distillation approach that aligns student learning with the end–task objectives for improved performance. To address the limitation of conventional knowledge distillation–where the student may learn generic features not optimized for the end task, we employ Task–Specific Knowledge Distillation, where the teacher model is trained on a particular task, allowing the student model to capture domain–relevant knowledge effectively. This targeted approach enhances the student model's performance beyond conventional knowledge distillation techniques.

Thus, our proposed methodology aims to construct a privacy–preserving model with enhanced task–specific performance. The process begins with preparing a general teacher model, specifically a pre–trained BERT model fine–tuned on data similar to the target dataset. Next, knowledge distillation is applied to derive a general student model, capturing essential knowledge in a compressed form. The general teacher model is then fine–tuned using Differen-

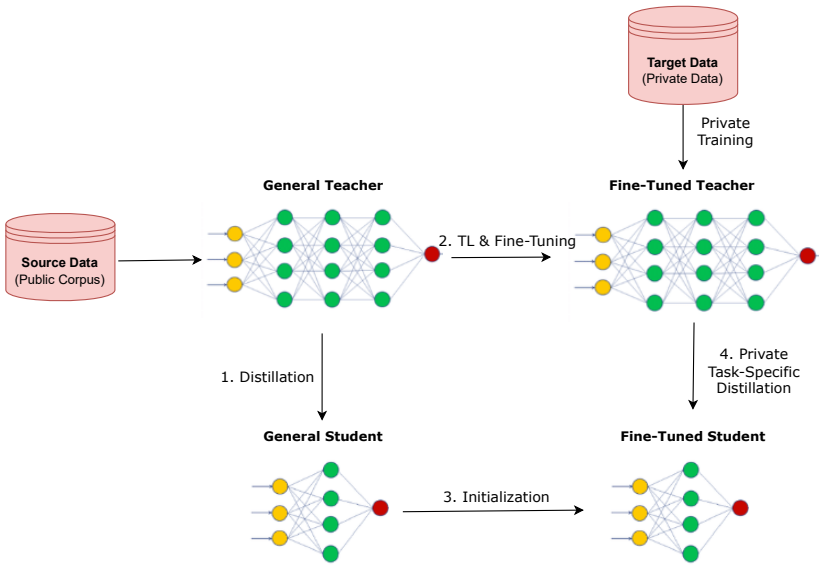


Figure 5.2: Overview of *TSKD*: Task–Specific Knowledge Distillation with Differential Privacy framework. Each step of the framework is elaborated in corresponding sections of our methodology (e.g., Step 1 in Section 5.3.1)

tially Private Stochastic Gradient Descent (DPSGD) with task–specific data, ensuring adaptability while preserving privacy guarantees. Following this, the student model is initialized with the parameters of the distilled general student model. Finally, task–specific knowledge distillation is performed, leveraging DPSGD to transfer distilled knowledge from the fine–tuned teacher model to the initialized student model. This iterative process enables the student model to acquire task–specific knowledge while maintaining strong privacy protection, making it suitable for privacy–sensitive applications. The detailed steps of this methodology are elaborated in subsequent sections, with a step–by–step procedure outlined in Algorithm 12. The notations used in the Algorithm 12 are summarized in Table 5.1.

5.3.1 Preparation of General Teacher Model

The first step involves preparing a general teacher model, which serves as the foundation for subsequent stages. We select a pre–trained BERT model fine–tuned on a task similar to the target task of interest. This model, referred to as the general teacher model, leverages transfer learning by encapsulating valuable knowledge from a related domain. Next, we perform knowledge distillation to derive a general student model, which compresses the essential knowledge and patterns learned by the general teacher model into a more efficient representation. This step is illustrated in Step 1 of Figure 5.2.

Algorithm 12 *TSKD*: Task–Specific Knowledge Distillation with DP

Require: Pre–trained teacher model θ_{teacher}, task–specific data, ϵ, δ
Ensure: Student model $|\theta_{\text{student}}| < \gamma \cdot |\theta_{\text{teacher}}|$ & θ_{student} are (ϵ, δ)–DP

Step 1: General Teacher Model

Take a pre–trained teacher model with θ_{teacher} on source task
$\theta_{\text{general\_student}} \leftarrow \text{Distill}(\theta_{\text{teacher}})$

Step 2: Private Fine Tuning

for each mini–batch X_i in task–specific data **do**
    Compute gradients: $\nabla L(\theta_{\text{teacher}}, X_i)$
    Clip gradients: $\nabla_{\text{clipped}} \leftarrow \text{clip}(\nabla L(\theta_{\text{teacher}}, X_i), \text{clip\_value})$
    Add noise: $\nabla_{\text{noisy}} \leftarrow \nabla_{\text{clipped}} + \text{noise}(\epsilon_1, \text{noise\_scale})$
    Update parameters: $\theta_{\text{teacher}} \leftarrow \theta_{\text{teacher}} - \text{learning\_rate} \times \nabla_{\text{noisy}}$
end for

Step 3: Initialization of Student Model

$\theta_{\text{student}} \leftarrow \theta_{\text{general\_student}}$

Step 4: Private Task Specific Distillation

for each mini–batch X_i in task–specific data **do**
    Compute gradients: $\nabla L(\theta_{\text{student}}, X_i)$
    Clip gradients: $\nabla_{\text{clipped}} \leftarrow \text{clip}(\nabla L(\theta_{\text{student}}, X_i), \text{clip\_value})$
    Add noise: $\nabla_{\text{noisy}} \leftarrow \nabla_{\text{clipped}} + \text{noise}(\epsilon_2, \text{noise\_scale})$
    Update parameters: $\theta_{\text{student}} \leftarrow \theta_{\text{student}} - \text{learning\_rate} \times \nabla_{\text{noisy}}$
end for

5.3.2 Private Fine–tuning of General Teacher Model

After selecting the general teacher model, it undergoes fine–tuning to enhance its performance and adaptability for the target task. Since the target task involves sensitive personal data, it is crucial to fine–tune the model while preserving privacy. To achieve this, we utilize DPSGD optimizer, which ensures privacy protection by applying gradient clipping and noise addition during parameter updates. These mechanisms control the influence of individual data points and introduce stochastic noise, thereby safeguarding privacy. The fine–tuning process requires a privacy budget of (ϵ_1, δ_1) from the total budget (ϵ, δ). This step is visually represented in Step 2 of Figure 5.2. By fine–tuning with DPSGD, the model effectively adapts to the target task while maintaining strong privacy guarantees, achieving a balance between utility and privacy preservation.

5.3.3 Initialization of Student Model

After fine–tuning the teacher model, the student model is initialized using the parameters obtained from the previously distilled general student model. Since this initialization is derived from a publicly pre–trained model, the associated privacy cost is zero. This step is visually represented in Step 3 of

Figure 5.2. The initialization serves as a crucial foundation for the subsequent task–specific knowledge distillation process. By leveraging knowledge distilled from the teacher model, the student model inherits essential insights and patterns, providing a well–informed starting point for further refinement and adaptation to the target task. While prior research has predominantly focused on initializing the student model with the teacher model's weights to improve accuracy [Mir+22], in our approach–initializing from the distilled student model demonstrates superior performance.

5.3.4 Private Task–Specific Knowledge Distillation

The final step involves private task–specific knowledge distillation, where the fine–tuned teacher model transfers its distilled knowledge to the initialized student model, as illustrated in Step 4 of Figure 5.2. To ensure private training, we employ DPSGD allowing the student model to learn from task–specific data while maintaining privacy guarantees. This process consists of computing gradients, clipping them to prevent large updates, adding noise for privacy preservation, and iteratively updating the student model's parameters. The privacy cost incurred during this step is (ϵ_2, δ_2) from the total budget. By distilling task–specific knowledge from the fine–tuned teacher model under differential privacy constraints, the student model effectively adapts to the target task while adhering to strict privacy–preserving protocols.

5.3.5 Privacy Analysis

We now conduct a privacy assessment of our framework.

Theorem 4. *The student model parameters $\theta_{student}$ obtained from Algorithm 12 are (ϵ, δ)–DP with $\epsilon = \epsilon_1 + \epsilon_2$ and $\delta = \delta_1 + \delta_2$. Here (ϵ_1, δ_1) are the privacy parameters consumed in Step 2 and (ϵ_2, δ_2) are those consumed in Step 4 of the Algorithm.*

Proof. Our aim is to ensure that the Algorithm 12 guarantees (ϵ,δ)–DP for the sample data **D**. In the first step, the teacher model is a pre–trained model from a public dataset. So, it doesn't incur any privacy loss. In the next step, the teacher model is privately fine–tuned with access to the private database. It upholds a privacy budget of (ϵ_1,δ_1)–DP. In step 3, student model is initialized from open domain models, so it doesn't introduce an additional privacy loss. Finally, in the last step task–specific distillation is performed using the private database and the previously trained private teacher model from Step 2. This step fine-tunes the student model and requires an additional privacy budget of (ϵ_2,δ_2)–DP, since it involves accessing the private data again. Thus, the resultant student model parameters θ_{student} distilled from teacher model are privacy preserving with a privacy budget of (ϵ,δ) obtained via a composition of (ϵ_1,δ_1) and (ϵ_2,δ_2).

This completes the proof. □

Table 5.1: Notations

θ_{teacher}	teacher model parameters
θ_{student}	student model parameters
ϵ	privacy loss parameter
δ	probability of privacy guarantee being violated
γ	scaling factor (level of compression)
(ϵ_1, δ_1)	privacy budget for private fine–tuning of teacher model
(ϵ_2, δ_2)	privacy budget for task–specific knowledge distillation

5.4 Experimental Setup

In this section, we first discuss the datasets we used, the baseline models for comparison, the privacy budget and hyper–parameters of our methodology.

5.4.1 Source and Target Data

We describe the datasets and task that are considered for experimentation. We have used GLUE benchmark [Wan+18] for our evaluation that is an evaluation benchmark designed to measure the performance in NLP. Many existing studies including [Li+21; Mir+22] used them in their frameworks. We used few of the following tasks from GLUE: SST–2 (Single–Sentence text classification task), RTE (Recognizing textual entailment), QNLI (Question–answering NLI), COLA (Corpus of Linguistic Acceptability) and MRPC (Microsoft Research Paraphrase Corpus). Our selection spans a diverse array of tasks, varying in the number of training examples. Notably, QNLI boasts the largest training set with 104K examples, whereas RTE contains 2K examples, representing the smallest training dataset in our experimentation.

Our methodology incorporates transfer learning [PY09], where a model is first pre–trained on a source task and then fine–tunes it for the target task. This approach facilitates the transfer of information, patterns, and representations learned from the source data to the new training process, improving efficiency and performance. Transfer learning can be applied in both same–domain and cross–domain scenarios. In same–domain transfer learning, the source and target domains are closely related, whereas cross–domain transfer learning is used when limited knowledge about the target domain is available. Given that we had sufficient knowledge about the target task, we opted for same–domain transfer learning, specifically Inductive Transfer Learning. In this approach, while the source and target domains remain similar, the tasks performed may differ, allowing the model to leverage domain–specific knowledge effectively.

Initially, we utilize pre–trained models that have been fine–tuned on tasks similar to the downstream target task. Selecting an appropriate source task for pre–training is critical to achieving optimal performance in the target task, as it ensures that the features learned during pre–training are relevant and trans-

ferable. For example, when working with QNLI (a large question–answering dataset) as the target task, we chose SST–2 (a sentence classification task) as the source task for pre–training. Similarly, for SST–2 as the target task, we selected MRPC (Microsoft Research Paraphrase Corpus) as the pre–training task. For RTE (Recognizing Textual Entailment) as the target task, we used WNLI (a reading comprehension task) as the source task. Pre–training on a dataset from the same domain or a closely related domain allows the model to capture domain–specific nuances, which can significantly enhance its performance on the target task. By leveraging large–scale and diverse datasets for pre–training, the model learns rich linguistic representations that can be effectively transferred to downstream tasks, improving generalization and robustness.

5.4.2 Baselines

We conducted a comprehensive comparison of our proposed methodology with several baseline models. The baseline models include state–of–the–art models, such as the BERT–base [Dev18], which consists of 12 transformer layers, the BERT–tiny model [Tur+19] with 6 transformer layers, and the DistilBERT model [San+19], which is a distilled version of BERT–base. All of these pre–trained models were fine–tuned on our target tasks under both privacy–preserving and privacy–agnostic setups. This approach ensures a fair comparison between our methodology and baseline models, including those that do not incorporate privacy–preserving mechanisms.

5.4.3 Privacy Budget and Hyper–parameters

To compare our work with the closely existing work [Mir+22], we adopted the same privacy budget as they used i.e., $\epsilon = 1$, and $\delta = \frac{1}{N}$, where N is the number of samples in the dataset. We allocated a privacy budget of (ϵ_1, δ_1) for fine–tuning the teacher model, followed by (ϵ_2, δ_2) for task–specific knowledge distillation. The overall privacy budget utilized was $(\epsilon, \delta) = (1, \frac{1}{N})$. We experimented with a learning rate of 10^{-5}, batch size = 64, maximum epochs = 10. The rate of compression in knowledge distillation is controlled by two parameters: α (weighting factor) and *temperature* (softmax temp). α provides a balance between student's own loss and distillation loss (KL loss). Higher alphas put more emphasis on mimicking teacher's prediction, whereas lower alphas give more weight to student's own prediction. In our experiments, we used $\alpha = 0.5$. Parameter *temperature* controls softening the teacher's probability before it is used to train the student. Higher value leads to softer probability distribution making training less reliant on hard targets, whereas lower values leads to sharper probability. We used temperature value of 5 in our experiments. The experimentation were performed on Google Colab with Intel Xeon CPU and 13GB of RAM. The GPU used was NVIDIA Tesla K80 with 12 GB of VRAM.

Table 5.2: Comparison of our approach *TSKD* with 12–layer BERT (BERT–base) and 6–layer BERT (BERT–tiny) which are fine–tuned using DPSGD with privacy budget ϵ =1.

Model	Training	Teacher	Num-Params	SST–2	RTE	QNLI	COLA	Avg
BERT–base	DP–Finetune	-	109M	92.20	59.56	87.8	81.30	80.21
BERT–tiny	DP–Finetune	-	5M	78.89	55.23	81.42	69.12	71.16
BERT–tiny	*TSKD*	BERT–base	5M	86.23	59.92	83.17	72.38	75.42

5.5 Results and Discussion

We now present the results obtained from our methodology and provide a detailed discussion on their implications.

5.5.1 A Comparative Analysis with Differentially Private Fine–tuned Models

We conducted experiments on our proposed approach, Task–Specific Knowledge Distillation (*TSKD*), using BERT–base as the general teacher model, which was subsequently distilled into a general student model (BERT–tiny). The evaluation was performed on four different datasets, as previously described, to measure model accuracy. To assess the effectiveness of *TSKD*, we compared its performance against pre–trained baseline models–BERT–base and BERT–tiny, both fine–tuned using DPSGD on the four target datasets. This comparison aimed to determine how well our proposed compression approach preserved accuracy while ensuring privacy. For a fair comparison with existing studies, we used a privacy budget of $\epsilon = 1$ for fine–tuning both the baseline models and our *TSKD* approach. A lower privacy budget provides stronger privacy guarantees but comes at the cost of reduced model utility.

The results are summarized in Table 5.2. When fine–tuning the BERT–base model with DPSGD, we achieved an accuracy of 92.2% on the SST–2 dataset. In contrast, our proposed *TSKD* approach attained an accuracy of 86.23%, demonstrating a slight trade–off in performance. Whereas, when fine–tuning the BERT–tiny model (which also serves as our student model) using DPSGD with the same privacy budget, we obtained an accuracy of 78.89% on SST–2 dataset. This analysis highlight that *TSKD* significantly reduces model size–from 109 million parameters (BERT–base) to 5 million parameters–while sacrificing only 6% accuracy compared to the privacy–preserving fine–tuned BERT–base model. This demonstrates the efficacy of our approach in achieving a trade–off between model compression, privacy preservation, and performance.

Table 5.3: Comparison of our approach *TSKD* with BERT–base and BERT–tiny which are fine–tuned without preserving privacy

Model	Training	Teacher	Num-Params	SST–2	RTE	QNLI	COLA	Avg
BERT–base	Finetune	-	109M	92.31	61.01	87.9	80.72	80.48
BERT–tiny	Finetune	–	5M	79.93	58.48	81.96	69.12	72.37
BERT–tiny	*TSKD*	BERT–base	5M	86.23	59.92	83.17	72.38	75.42

Table 5.4: Comparison of our approach *TSKD* with 6–layer DistilBERT which is fine–tuned without and with DP

Model	Training	Teacher	Num-Params	SST–2	RTE	QNLI	COLA	Avg
DistilBERT	Finetune	-	66M	90.90	59.12	87.33	81.49	79.71
DistilBERT	DP–Finetune	-	66M	90.71	58.12	88.46	77.08	78.59
DistilBERT	*TSKD*	BERT–base	66M	89.90	60.35	84.29	68.16	75.67

5.5.2 A Comparative Analysis with Fine–tuned Models in a Privacy–Agnostic Context

We also conducted a comparative analysis of our proposed approach against non–private models. Privacy–preserving techniques often introduce a trade–off between privacy and utility. In scenarios where maximizing the model utility is the primary objective, privacy constraints may not be desirable. To assess this trade–off, we fine–tuned the same pre–trained models, specifically BERT–base, on downstream tasks without applying any privacy–preserving techniques. This resulted in an accuracy of 92.31% on the SST–2 dataset, which is marginally higher than the accuracy achieved with the differentially private fine–tuned model. In contrast, our proposed approach, *TSKD*, achieved an accuracy of 86.23%. This indicates that by sacrificing only 6% of model accuracy, one can obtain a differentially private model. These results, summarized in Table 5.3, which also aligns with prior research findings, which suggest that large pre–trained models fine–tuned using DPSGD can achieve performance comparable to non–private models [Mir+20; Meh+22].

5.5.3 Initialization of Student Models with Pre–Distilled Models

We further examined whether improved initialization of student models enhances performance. In this experiment, as presented in Table 5.4, we initialized the student model with a pre–distilled DistilBERT model instead of BERT–tiny. When this initialized student model was incorporated into our *TSKD* approach, we observed an improvement in model accuracy, achieving 89.90%. To provide a comprehensive comparison, we also directly fine–tuned the DistilBERT model on the target tasks, both with and without DPSGD. The resulting accuracies were 90.90% (without DPSGD) and 90.71% (with

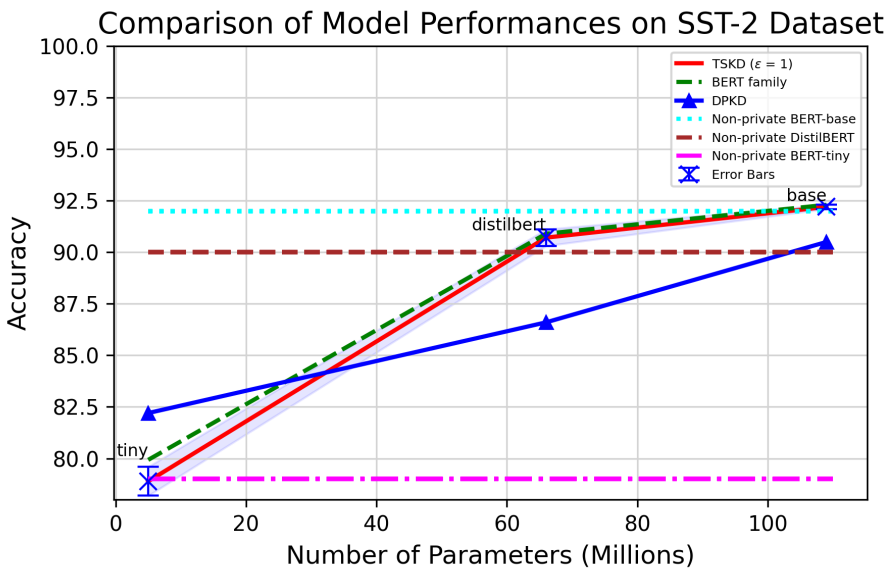


Figure 5.3: Comparison of Model Performance on SST–2 Dataset: We plot the accuracy of BERT–base, DPKD, and our *TSKD* approach ($\epsilon = 1$) against the number of parameters.

DPSGD), respectively. These results indicate that our proposed approach *TSKD* with DP achieves performance comparable to non–private learning while maintaining strong privacy guarantees. Additionally, our method performs competitively against standard baseline models, such as BERT–base, highlighting its effectiveness in privacy–preserving model compression.

The Figure 5.3 illustrates the performance of various models on the SST–2 dataset concerning accuracy versus the number of parameters. Three primary models are compared: BERT family models, DPKD [Mir+22], and our proposed approach *TSKD* with ϵ equal to 1. Each model's accuracy is plotted against the number of parameters, showcasing their relative performance. Error bars represent the variability in accuracy for our model. Additionally, three specific points on the BERT–base curve are labeled as BERT–tiny, DistilBERT, and BERT–base, indicating performance benchmarks. It depicts that our approach is significantly better than existing works, and even comparable to non–private state–of–the–art models.

5.6 Revisiting Model Compression: Beyond KD

The proposed *TSKD* approach achieves performance comparable to both private and non–private baselines while significantly reducing the model size by 95%. However, there remains a scope for further improvement. Firstly, this approach is task–specific. While it demonstrates effectiveness across multiple tasks, it may not generalize well to all tasks, as it is not a universally applicable method. Developing a more generalized framework that adapts to a wider range of tasks could enhance its applicability. Secondly, prior research suggests that model pruning can, in some cases, yield better performance than knowledge distillation [Mir+22]. Given this insight, we now explore the potential benefits of pruning as an alternative to this method.

We specifically focus on structural pruning, which involves removing entire filters or structural components from the neural network. Unlike unstructured pruning, which removes individual weights and often requires specialized hardware for efficient execution, structural pruning offers better hardware efficiency and is more suitable for deployment on real–world systems [HX23]. It is particularly effective for compressing large networks where significant reductions are needed, whereas unstructured pruning is more appropriate for smaller models requiring fine–grained adjustments without altering the overall structure. Structural pruning can be achieved through various techniques, including L1–based pruning [Han+15], which eliminates the parameters based on their magnitude, first–order importance estimation [Hou+20], which ranks components based on their contribution to the loss function, Hessian–based techniques [Kur+22], which consider second–order information to determine pruning importance, and the Optimal Brain Surgeon method, which removes parameters based on their impact on the network's overall loss. Different studies have explored various pruning strategies, targeting different units of the model,

such as entire layers, multi–head attention mechanisms, or feed–forward layers. We chose structural pruning because it aligns well with hardware–friendly optimizations, ensuring that the pruned model remains computationally efficient and easier to deploy on resource–constrained environments. Furthermore, structural pruning results in models that are easier to fine–tune post–pruning, maintaining a balance between efficiency and task performance.

Pruning can also be thought of as analyzing the notion of redundancy in the transformer models, which could be layer–level redundancy or neuron–level redundancy [Dal+20]. LLM–Pruner [MFW23] computed the importance of channel–wise weights to perform structural pruning and then fine–tuned the pruned model using LoRA. However, each channel may contain crucial information and pruning them can degrade the performance. Also, they didn't consider if any sensitive data was used in training, and its privacy implications. Sparse–GPT [FA23] performed unstructured pruning on weights, while compensating for weights that are not pruned. They performed one–shot pruning, but many research works demonstrate that fine–tuning with LoRA saves computational time and efficiency. LLM–Streamline approach [Che+24] performs pruning and removes redundant layers by computing cosine similarity and then a lightweight network is trained to replace the pruned layers. However, we show that fine–tuning is more efficient than re–training, and also no privacy parameters were considered, which is an essential component for our approach.

However, pruning techniques present several challenges, some of which we aim to address in the new approach. These challenges include:

Complexity of Optimal Pruning. Determining which weights or neurons to prune is complex. Strategies like magnitude–based pruning, where weights with small magnitudes are removed, might not always capture the most important parameters. More sophisticated techniques, such as those based on sensitivity analysis or learned pruning, require additional computational resources and can be more difficult to implement.

Overhead of Post–Pruning. Following model pruning, it is often necessary to retrain or fine–tune the model to restore any lost performance. Also, privacy is still a critical issue, as models can memorize and unintentionally expose sensitive information from the training data. While the literature has largely overlooked privacy–preserving strategies during this phase, our approach addresses this gap by exploring effective techniques for secure and private fine–tuning after pruning.

5.7 Approach 2: *PrunePrivyTune* With DP

We now present our methodology: *PrunePrivyTune*. We divide it into three main components: efficient pruning, private fine–tuning and data synthesis with the fine–tuned model. We conceptualize pruning as the removal of redundant layers in the model using pairwise cosine similarity. Firstly, we extract

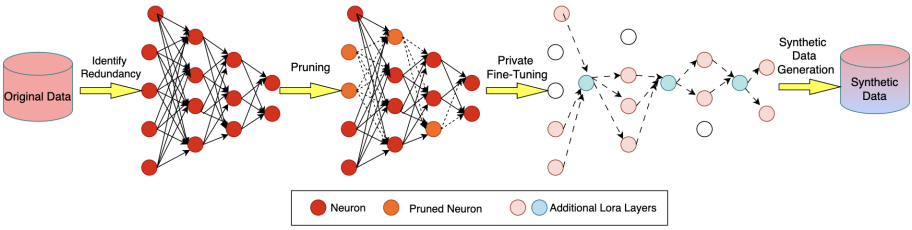


Figure 5.4: *PrunePrivyTune*: An Approach For Efficient Pruning and Private Fine–Tuning

the hidden states of the model, and compute the pairwise cosine similarity between these states. We then find the average pairwise cosine similarity and prune layers based on it. This leads to information loss, so to mitigate the potential loss of information, the model should be either re–trained or fine–tuned. Given that re–training can be computationally intensive, we opt for fine–tuning, and we fine–tune the pruned model with the task of interest in a privacy–preserving manner, using DPSGD. Now, we generate the synthetic data with the fine–tuned model to assess privacy vulnerabilities. We perform training data extraction attack and assess whether the model memorized training data and has the capability to regenerate it or not. The methodology is diagrammatically depicted in Figure 5.4 with step by step explanation described in the next sub–sections.

5.7.1 Pruning

The objective of pruning is to minimize the redundancy in the model while maintaining similar performance. The idea is that if two consecutive layers are very similar, one of them might be redundant and can be pruned. This method emphasizes incremental knowledge changes across layers. Let L be the set of all layers in the model. For each pair of consecutive layers $(\ell, \ell+1)$, the cosine similarity is denoted by cos-sim($x^{(\ell)}$,$x^{(\ell+1)}$), where $x^{(\ell)}$ denotes the ℓ-th hidden state. The objective function is to minimize the sum of cosine similarities for the pruned layers, subject to the constraint that the pruned model maintains a performance metric $\mathcal{P}$ within a threshold Δ of the original model's performance $\mathcal{P}_0$. Formally, the optimization problem can be written as:

$$\begin{aligned} \min_{S \subseteq \mathcal{L}} \quad & \sum_{\ell \in S} \frac{1}{2} \left(\text{cos-sim}(x^{(\ell)}, x^{(\ell+1)}) + \text{cos-sim}(x^{(\ell)}, x^{(\ell-1)}) \right) \\ \text{subject to} \quad & \mathcal{P}(S) \geq \mathcal{P}_0 - \Delta \end{aligned} \tag{5.1}$$

where S is the set of pruned layers, $\mathcal{P}(S)$ is the performance of the model after pruning the layers in S, $\mathcal{P}_0$ is the original model's performance and Δ is the

allowable degradation in performance. The step by step explanation of the proposed methodology for pruning is found as follows.

Extract Hidden States

LLM mainly use a transformer architecture, which is made up of several transformer encoder–decoder layers. The effect of each layer can be analyzed as a transformation of the previous hidden state. Importantly, these layers follow a residual structure, meaning that instead of replacing the input with a new value, each layer adds a learned transformation to the input, helping with gradient flow and stability during training. The transformation performed at layer ℓ can be expressed as:

$$x^{(\ell+1)} = x^{(\ell)} + f(x^{(\ell)}, \theta^{(\ell)}) \tag{5.2}$$

In this equation, layer $f(.)$ represents a transformation function applied at a layer, and $\theta^{(\ell)}$ denotes the parameters of layer ℓ that define how f transforms the input. Therefore, we can understand the importance of each layer in LLMs by looking at how much it changes the input hidden states.

Compute Pairwise Cosine Similarity

To quantify the redundancy between two layers in a model, we compute the cosine similarity between their corresponding hidden states (x, x'), which is computed as follows:

$$\text{cos-sim}(x, x') = \frac{\sum_{i=1}^{L}\sum_{j=1}^{d} x_{i,j} \cdot x'_{i,j}}{\sqrt{\sum_{i=1}^{L}\sum_{j=1}^{d}(x_{i,j})^2} \cdot \sqrt{\sum_{i=1}^{L}\sum_{j=1}^{d}(x'_{i,j})^2}} \tag{5.3}$$

This metric captures the angular similarity between two activation matrices, treating them as flattened vectors. A high cosine similarity indicates that the two layers produce highly aligned representations, suggesting redundancy and potential for pruning.

Identification of Redundant Layers

We define the *redundancy metric* to evaluate how similar each layer is to its neighboring layers. To do this, we compute pairwise cosine similarity of each layer ℓ and its adjacent layers $\ell-1$ and $\ell+1$. For a given layer ℓ, the average cosine similarity is calculated as follows.

$$\text{avg\_cos}_\ell = \frac{1}{2}\left(\text{cos-sim}(x^{(\ell-1)}, x^{(\ell)}) + \text{cos-sim}(x^{(\ell)}, x^{(\ell+1)})\right) \tag{5.4}$$

The average cosine similarity quantifies how similar the transformations are between a given layer and its adjacent layers. A higher value indicates that the layer has a higher degree of similarity with its neighbors, suggesting potential

redundancy. Based on this, we define *threshold–based pruning* strategy. To determine which layers to prune, set a threshold value τ for the average cosine similarity. Layers with an average similarity above this threshold are considered redundant. That is,

$$\text{redundant layers} = \{\ell \mid \text{avg\_cos}_\ell > \tau\} \tag{5.5}$$

where τ is the predefined threshold value. These identified layers are considered candidates for pruning, as they offer limited unique transformation and can be removed with minimal impact on the model's overall capacity.

Prune Redundant Layers

In this step, the pruning procedure is performed by removing the redundant layers identified based on the computed average cosine similarity and adjusting the connections between the preceding and succeeding layers to maintain the model's structural integrity. The proposed pruning method can be executed by following the steps described above, as detailed in Algorithm 13. After pruning, the model undergoes fine-tuning to recover potential performance loss due to layer removal. This step ensures that the compressed model retains predictive capability while benefiting from reduced complexity and improved efficiency.

5.7.2 Private Fine–Tuning

After pruning, the model may suffer from information loss and performance degradation. To mitigate this, the model needs to be either re–trained or fine–tuned. Fine–tuning is generally preferred over re–training because it is computationally more efficient and leverages pre–learned features, resulting in faster convergence. Several fine–tuning techniques allow updating pre–trained models without modifying all the weights. One widely used method is LoRA (Low–Rank Adaptation) [Hu+21], which introduces low–rank decomposition matrices within each dense layer. This approach significantly reduces the number of trainable parameters compared to full fine–tuning while maintaining model effectiveness. Due to its efficiency and scalability, LoRA is the preferred technique for our fine–tuning process.

However, fine–tuning carries the risk of memorizing the training data, as models can inadvertently learn and store specific data points. To mitigate this risk, differential privacy can be applied, which helps prevent data memorization and reduces the potential for privacy leakage. Since fine–tuning is performed on sensitive target data, incorporating DP is essential to ensure privacy protection. To address this, we propose DP–LoRA fine–tuning, a method for privately fine–tuning the pruned model on sensitive target tasks. LoRA is particularly effective in this setting because it updates only a small subset of parameters instead of fine–tuning the entire model. Specifically, instead of updating the full weight matrix W, LoRA introduces low–rank matrices L and R such that the updated weight matrix becomes: $W + LR$. During fine–tuning, only the

matrices L and R are updated, significantly reducing the number of trainable parameters. This improves computational efficiency, especially when combined with DPSGD.

In prior research on privacy–preserving training of LLMs, the entire set of weights were typically updated under DP constraints [Mir+22]. However, this approach is computationally expensive and often results in suboptimal performance when used with DPSGD. In contrast, fine–tuning a subset of parameters with DPSGD is generally more effective than re–training the full model with DP. There is limited research on privacy–preserving fine–tuning with LoRA, highlighting the novelty and potential advantages of our approach. The step–by–step methodology of DP–LoRA fine–tuning is detailed in Algorithm 14. Unlike traditional LoRA, which updates only the low-rank matrices L and R, our approach additionally applies gradient clipping and Gaussian noise to these updates, limiting the influence of individual training samples and providing formal privacy guarantees. This makes the fine–tuning process privacy–preserving while still benefiting from LoRA's parameter efficiency. Furthermore, by focusing on low–rank updates, our method achieves a balance between privacy protection and model efficiency, making it well–suited for large–scale models. This approach significantly reduces the computational overhead compared to traditional DP methods, without sacrificing model performance.

5.7.3 Data Synthesis using the Fine–Tuned LLM

Transformer models are known for their tendency to memorize the training data, making the application of DP essential to mitigate the risk of unintended data leakage. After fine–tuning the model with DP, we assess privacy risks by generating synthetic data and evaluating whether the fine–tuned model memorizes and inadvertently regenerates its training data. While auto–regressive language models are commonly used for text generation, recent advancements have demonstrated the effectiveness of masked language models (MLMs), such as BERT. In this approach, text is generated by iteratively predicting masked tokens and refining outputs. Specifically, MLMs predict masked tokens in an input text, and least likely predictions are re–masked and refined in successive iterations [Gha+19]. This technique has shown promising results in machine translation and other NLP tasks.

Inspired by this methodology, we employ a fine–tuned BERT model as MLM for synthetic data generation. The process begins by selecting input sentences from the target task and randomly masking a portion of the tokens. The model then predicts these masked tokens using top–k sampling, which introduces diversity by selecting one of the top–k most probable tokens for each masked position. Least likely predictions are iteratively re–masked and refined until all masked tokens are replaced, yielding a complete sentence. By repeating this process across a large number of sentences, a diverse and high–quality synthetic dataset is generated. To assess the privacy risks associated with the

Algorithm 13 Pruning Redundant Layers in a Model

Require: Model M, DataLoader D, Threshold τ, Performance Function P, Max Degradation Δ

Extract Hidden States(M, D):

Initialize hidden states $\mathsf{H} = [\ ]$
for each batch in D:
  $x \leftarrow$ batch[input_ids]
  outputs $\leftarrow M(x,$ output_hidden_states=True)
  hidden_states $\leftarrow$ outputs_hidden_states
  Append hidden_states to H
end for
Return H

Compute Cosine Similarity(H):

Initialize pairwise cosine similarity $\mathsf{C} = [\ ]$
for sample in H
  for $\ell = 1$ **to** $L - 1$
    Append cos-sim$(x^{(\ell)}, x^{(\ell+1)})$ to $\mathsf{C}[\ell]$
  end for
end for
Return C

Identify Redundant Layers(C, τ):

Initialize redundant_layers $= \emptyset$
for $\ell = 1$ **to** L
  avg_cos$_\ell = \frac{1}{2}\left(\text{cos-sim}(x^{(\ell-1)}, x^{(\ell)}) + \text{cos-sim}(x^{(\ell)}, x^{(\ell+1)})\right)$
  If avg_cos$_\ell > \tau$ then
    Add ℓ to redundant_layers
  end if
end for
Return redundant_layers

Prune Layers(M, C, τ, P, Δ):

Initialize $\mathsf{P}_0 \leftarrow P(M)$, pruned layers $\mathsf{S} = [\ ]$
for $\ell = 1$ **to** $L - 1$
  If $\mathsf{C}[\ell] > \tau$ then
    Prune layer $\ell + 1$ from M, append to S
    $\mathsf{P}_{new} \leftarrow P(M)$
    If $\mathsf{P}_{new} < \mathsf{P}_0 - \Delta$ then
      Restore layer $\ell + 1$ to M, remove from S, BREAK
    end if
  end if
end for
Return M, S

main():

$H \leftarrow$ Extract Hidden States (M, D)
$C \leftarrow$ Compute Cosine Similarity (H)
redundant_layers $\leftarrow$ Identify Redundant Layers (C, τ)
Return Prune Layers(M, C, τ, P, Δ)

Algorithm 14 DPSGD LoRA Fine–Tuning

Require: Weight matrix W, Low–rank matrices L and R, η, Noise scale σ, Clipping norm C, Dataset $\mathcal{D}$, ϵ

1: Initialize L and R with small random values
2: **for** each minibatch $B \subset \mathcal{D}$ **do**
3: Compute gradients for L and R:

$$\nabla L_B, \nabla R_B \leftarrow \frac{1}{|B|} \sum_{i \in B} \nabla L_i, \nabla R_i$$

4: Clip gradients to norm C:

$$\nabla L_B \leftarrow \frac{\nabla L_B}{\max(1, \frac{\|\nabla L_B\|_2}{C})}, \quad \nabla R_B \leftarrow \frac{\nabla R_B}{\max(1, \frac{\|\nabla R_B\|_2}{C})}$$

5: Add noise to the gradients:

$$\tilde{\nabla} L_B \leftarrow \nabla L_B + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}), \quad \tilde{\nabla} R_B \leftarrow \nabla R_B + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$$

6: Update L and R using noisy gradients:

$$L \leftarrow L - \eta \tilde{\nabla} L_B, \quad R \leftarrow R - \eta \tilde{\nabla} R_B$$

7: **end for**
8: **Return** updated weight matrix $W + LR$

synthetic data, we compare it with the real dataset using training data extraction attacks. This evaluation helps determine whether the fine–tuned model inadvertently memorizes and regenerates sensitive training samples, ensuring that DP effectively mitigates privacy risks.

5.7.4 Privacy Analysis of *PrunePrivyTune*

We employ DP to safeguard sensitive data during model training, specifically using a DPSGD optimizer. The privacy guarantee in DP is characterized by the parameter ϵ. In our approach, the ϵ value directly corresponds to the ϵ used in DPSGD. A crucial aspect of our methodology is that each training example x_i is utilized only once during the training process. This is important because, in standard DP mechanisms, repeated use of the same data point leads to cumulative privacy loss, a concept known as composition. However, since each training example is used exactly once, there is no iterative exposure, and thus no need to account for composition. Consequently, the privacy budget of the whole process is also ϵ. Naturally, the same applies for privacy parameter δ. This approach is similar to Local Differential Privacy [Jos+18], where each user's data is independently protected before being incorporated into the model.

5.8 Privacy Risk Assessment

The AI models are vulnerable to various privacy attacks such as membership inference attacks [Sho+17], model inversion attacks [FJR15], and training data extraction attacks [Car+21]. We focus on evaluating the effectiveness of our privacy mitigation strategy through the lens of the training data extraction attack, as it is the most relevant and direct approach to assess our concerns. A training data extraction attack is effective to find whether a language model has memorized some portions of training data that could lead to privacy leakage. This attack aims to reconstruct exact instances from the training data, instead of producing similar or approximates instances. To provide a clear context, we define memorization within the scope of language models, the threat model, and evaluation of privacy attack as follows.

5.8.1 Memorization

To some extent, memorization is a natural byproduct of how language models are trained. Language models are trained using maximum likelihood estimation, where the objective is to predict the next word (or a missing word) by maximizing the probability of the correct answer given the training data. To achieve this, the model learns statistical patterns and relationships from the dataset. However, in doing so, the model may also store and recall specific training examples, especially if the data appears frequently or is overrepresented.

When a dataset contains repeated patterns, the model may memorize them to optimize its objective of maximizing likelihood and improving accuracy. However, a more concerning issue arises when the model memorizes rare or unique data points, especially if they contain sensitive information and appear only once in the training set. If the model unintentionally reproduces such unique samples, it indicates privacy leakage, as the model has learned something it was not intended to store or reveal.

5.8.2 Threat Model

We consider an adversary with black–box access to the model. This means the adversary can compute the probability of arbitrary sequences $f_\theta(x_1, \ldots, x_n)$ and generate text or obtain next–word predictions based on these probabilities. However, the adversary does not have access to the model's internal parameters, such as the individual weights or hidden states (e.g., attention vectors). This type of attack is highly realistic in real–world scenarios, especially considering that many language models, including GPT models, are often trained on sensitive data. Through careful prompt engineering, an adversary can generate synthetic text that may inadvertently reveal sensitive information. This risk highlights the importance of implementing robust privacy–preserving techniques during model training. The adversary's primary objective is to extract

specific instances of memorized training data from the model. The effectiveness of the attack is evaluated based on the sensitivity of the extracted information, with the assumption that more sensitive or unique examples represent a greater privacy risk. Consequently, the attack's strength is measured by the degree of privacy compromise, specifically focusing on how well the adversary can retrieve highly private or unique training examples

5.8.3 Privacy Attack Evaluation

We utilize two evaluation measures to quantify privacy. The first is a natural likelihood measure, the perplexity of a sequence, which quantifies how well the language model predicts the given data [Car+21]. Specifically, given a sequence of tokens $x_1, \ldots, x_n$, the perplexity P is defined as:

$$P = \exp\left(-\frac{1}{n}\sum_{i=1}^{n}\log f_\theta(x_i \mid x_1, \ldots, x_{i-1})\right) \tag{5.6}$$

For a sequence of tokens $x_1, x_2, \ldots, x_n$, the perplexity is computed by summing the log-likelihoods (i.e., the logarithm of the probability of each token x_i given its previous tokens) across the entire sequence. The average of these log-likelihoods is then taken, and the exponential of this value gives the perplexity. Perplexity captures the model's uncertainty regarding the sequence, where a lower perplexity indicates that the model assigns higher average probabilities to the tokens in the sequence, suggesting the model is less surprised by the sequence. This implies that the model has a stronger predictive capacity over the given data.

Another key metric is BERTScore [Zha+20], which uses pre–trained contextual embeddings from our fine–tuned model to measure the similarity between candidate and reference sentences via cosine similarity. BERTScore correlates well with human judgments at both sentence and system levels. It also calculates precision, recall, and F1 scores, providing a nuanced evaluation of language generation tasks.

5.9 Results and Discussion

We empirically evaluate the effectiveness of our approach *PrunePrivyTune*, analyzing its impact through different aspects. We emphasized the importance of pairwise cosine similarity metric that we used for our pruning in Section 5.9.1, We also compared the effectiveness of re–training in Section 5.9.2, the impact of pruning rate in Section 5.9.3, and the effect of DP in fine–tuning in Section 5.9.4, with a comparison between training and fine–tuning in Section 5.9.5, and a fair comparison with existing baselines in Section 5.9.6. We also discuss the advantages of redundancy based pruning in Section 5.9.7 and analyze our training data extraction attack in Section 5.9.8.

5.9.1 Significance of Pairwise Cosine Similarity

We used pairwise cosine similarity metric in our proposed approach for identifying redundant layers and then pruning them. It is essential to showcase the efficiency of pairwise cosine similarity over traditional cosine similarity. So we provide a fair comparison between them in Table 5.5. The cosine similarity between consecutive hidden states ℓ and hidden state $\ell + 1$ is computed for all layers. These similarity measures are then used to identify the least important layers in the model. Higher similarity values between two consecutive layers indicate minimal changes in information, suggesting that the layers are redundant. As a result, such layers can be pruned without significantly affecting the model's performance.

We experimented with four different tasks, as mentioned earlier, and evaluated model accuracy. The results in Table 5.5 demonstrate that pruning models using pairwise cosine similarity leads to improved accuracy compared to using just cosine similarity for pruning. Additionally, we analyzed the impact of privacy by comparing model performance with and without DP. Even when DP is applied, models pruned using pairwise cosine similarity exhibit significantly better accuracy than those pruned with regular cosine similarity. For experiments involving DP, we set the privacy budget to $\epsilon = 1$. The results show that models pruned using pairwise cosine similarity consistently achieve higher accuracy across all datasets compared to those pruned with standard cosine similarity. Furthermore, LoRA fine–tuning improves model performance relative to standard training, with the most significant improvements observed when combining pairwise cosine similarity pruning with LoRA fine–tuning. When DP is applied, this combination proves to be the most effective, as it maintains high accuracy while ensuring privacy protection. These results raise an intriguing question: **why does the accuracy of the tasks from our approach is higher even with DP than without it?** We hypothesize that this improvement stems from the sequential application of pruning, DP mechanisms, and fine–tuning. Pruning redundant layers using pairwise similarity first reduces the number of parameters, ensuring that only the most impactful features are retained. This simplification makes gradient clipping for DP more effective, as it focuses on controlling the magnitude of updates for meaningful parameters, preventing overly large updates that could dominate learning. With fewer gradients to update, the noise added during the DP step is distributed over a smaller set of critical parameters, reducing the effective noise per parameter. This targeted application of noise acts as a regularizer, further aiding in avoiding over–fitting to the training data. Finally, LoRA fine–tuning adapts the model within a low–rank subspace which limits the degree of freedom and ensures that the added DP noise minimally impacts meaningful updates. This combination of pruning, DP noise, and LoRA not only focuses the learning process on essential features but also facilitates more robust generalization, yielding better utility even under the constraints of DP. These results are also aligned with existing studies [SO17] that in some cases the prediction accuracy

Table 5.5: Comparison of cosine similarity vs Pairwise-cosine similarity

Pruning Strategy	Training	DP	SST-2	RTE	MRPC	COLA
Cosine	Train	-	88.75	62.45	80.21	77.93
Pairwise-Cosine	Train	-	91.05	64.62	82.84	78.14
Cosine	LoRA Fine-Tune	-	89.33	44.76	82.34	68.11
Pairwise-Cosine	LoRA Fine-Tune	-	91.97	47.29	83.71	69.12
Cosine	Train	Yes	88.72	46.93	60.65	62.48
Pairwise- Cosine	Train	Yes	90.11	49.09	63.33	64.21
Cosine	LoRA Fine-Tune	Yes	90.43	47.32	80.28	76.15
Pairwise-Cosine	LoRA Fine-Tune	Yes	92.31	49.45	82.37	79.70

improves because of the noise reduction effects of the condensation process. When ML models are resistant to errors, some noise addition does not reduce dramatically the accuracy of the model. in fact, adding noise may results into models that are better from the point of view of generalization.

Figure 5.5 shows the heatmap between different BERT layers. It presents the computation of pairwise cosine similarity between hidden states of layers of BERT which includes 12 transformer layers and a embedding layer. The color gradient of heatmap ranges from shades of red to blue, illustrating the degree of similarity. Color red indicates higher similarity between layers which suggests that consecutive layers are more alike, while color blue indicates lower similarity between layers, suggesting greater differences between layers. This visualization allows us to identify which layers produce similar representations and which layers exhibit distinct characteristics, providing insights into the internal structure and behavior of the BERT model across its various layers.

5.9.2 Comparative Analysis of Model Re–training: With and Without Differential Privacy

We evaluate the impact of full model retraining following our proposed pruning methodology. Instead of fine–tuning the pruned model, we retrain it from scratch to recover any lost information. While this approach can potentially restore model performance, it is significantly more computationally expensive compared to fine–tuning.

Table 5.6 presents the results of retraining the pruned model across different sparsity levels, ranging from 10% (sparsity = 0.1) to 30% (sparsity = 0.3). This evaluation helps analyze the trade–offs between model size, performance, and privacy. We further assess the impact of DP by comparing results with and without DP, using a privacy budget of $\epsilon = 1$ when DP is applied. As sparsity increases from 0.1 to 0.3, a greater proportion of layers are pruned, reducing model complexity and inference time. However, this also results in a slight decline in accuracy across different tasks. For instance, in the SST–2 dataset,

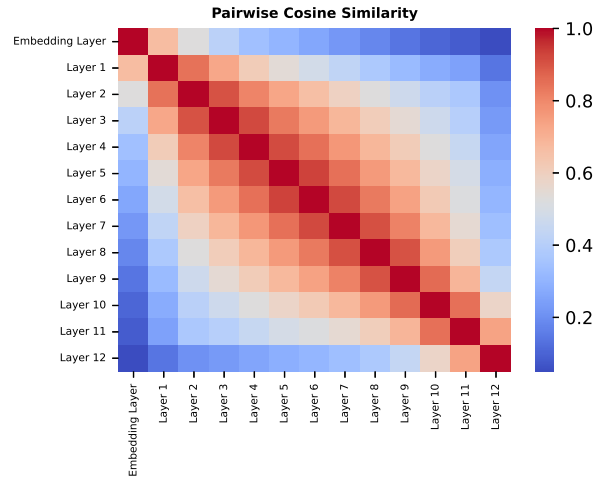


Figure 5.5: Pairwise Cosine Similarity between BERT layers

Table 5.6: Re–training the model with and without DP

Sparsity	DP	SST–2	RTE	MRPC	COLA	Avg
0.1	-	91.05	64.62	82.84	78.14	79.16
0.2	-	88.79	59.92	84.06	78.14	77.72
0.3	-	84.60	61.73	82.35	73.92	75.65
0.1	Yes	90.11	49.09	63.33	64.21	66.68
0.2	Yes	88.99	50.54	67.64	54.74	65.47
0.3	Yes	83.66	61.73	68.13	65.12	69.66

accuracy drops from 91.05% at 0.1 sparsity to 84.60% at 0.3 sparsity when retraining is performed without DP.

The introduction of DP further decreases accuracy beyond the effects of sparsity alone. Interestingly, the impact of DP combined with higher sparsity can be less severe compared to DP with lower sparsity levels. This suggests that higher sparsity might mitigate some of the negative effects of DP. For example, in the MRPC dataset, the accuracy with 0.1 sparsity and DP is 63.33%, while with 0.3 sparsity and DP, it improves to 68.13%. In summary, there is a trade–off between sparsity and model performance. Retraining the pruned model from scratch offers a more thorough recovery of accuracy, but it is more computationally demanding. Additionally, while DP reduces accuracy, its combined effect with higher sparsity may sometimes result in better performance than lower sparsity levels with DP.

5.9.3 The Effect of Pruning Rate on Accuracy

Higher sparsity levels result in models with fewer parameters, simplifying their structure and reducing the overall gradient magnitudes during training. This reduction enhances the efficiency of DP, as fewer gradients are subjected to clipping and noise addition. Notably, pruned models are inherently less sensitive to noise, allowing DP–induced noise to function as an effective regularizer rather than a source of distortion. By focusing updates on the most critical parameters, sparsity improves the signal–to–noise ratio in gradient updates, enabling the model to retain useful information even under DP constraints. Furthermore, pruning removes redundant parameters, reducing the risk of overfitting and encouraging the model to prioritize essential features. This synergy between sparsity and DP leads to a more robust training process, where higher sparsity levels help the model maintain its utility despite the privacy–preserving modifications. This positive correlation between pruning rate and model performance under DP is evident in Table 5.6 and aligns with findings from prior studies [AP23], which observed improved test accuracy as the pruning rate increased when models were trained with DP.

5.9.4 Comparative Analysis of Model Fine–Tuning: With and Without Differential Privacy

Table 5.7 highlights the impact of our proposed methodology on model accuracy by combining model compression with parameter–efficient fine–tuning. We applied the pruning strategy at various sparsity levels to reduce the model size and then fine–tuned the pruned models using the LoRA (Low–Rank Adaptation) method. The table captures the model's accuracy across different tasks, both when DP is applied (privacy–preserving) and when it is not (privacy–agnostic). This comprehensive evaluation demonstrates how model sparsity and privacy considerations influence the performance, providing insights into the trade–offs between efficiency, privacy, and accuracy. It can be observed that higher sparsity levels lead to reduced accuracy in both private and non–private models. However, the impact of increased sparsity is more severe in non–private models. Also, DP decreases model accuracy, but its impact is less severe at higher sparsity levels. The added noise for privacy protection contributes to accuracy loss, but this loss can be somewhat mitigated by higher sparsity. For example, in RTE dataset, the accuracy drops from 47.29% to 46.57% without DP while it just drops from 49.45% to 48.73% with DP. Trade–offs between sparsity, DP, and fine–tuning need to be carefully balanced based on the application requirements. Lower sparsity and better fine–tuning strategies (like LoRA) help in maintaining model accuracy while still achieving some level of pruning and privacy protection.

Table 5.7: Performance Comparison of Parameter–Efficient Fine–Tuning Using LoRA with and without DP

Sparsity	DP	SST–2	RTE	MRPC	COLA	Avg
0.1	-	91.97	47.29	83.71	69.12	71.69
0.2	-	89.33	46.93	83.21	68.42	71.97
0.3	-	86.69	46.57	78.38	68.05	71.25
0.1	Yes	92.31	49.45	82.37	79.70	75.95
0.2	Yes	89.44	46.57	80.72	78.25	73.74
0.3	Yes	85.32	48.73	82.11	76.63	73.20

Table 5.8: Training vs Parameter–Efficient Fine–Tuning using DP

Training	Sparsity	SST–2	RTE	MRPC	COLA
Train	0.1	90.11	49.09	63.33	64.21
LoRA Fine–Tune	0.1	92.31	49.45	82.37	79.70
Train	0.2	88.99	50.54	67.64	54.74
LoRA Fine–Tune	0.2	89.44	46.57	80.72	78.25
Train	0.3	83.66	61.73	68.13	65.12
LoRA Fine–Tune	0.3	85.32	48.73	82.11	76.63

5.9.5 Training vs Fine–Tuning

In addition to evaluating our pruning and fine–tuning strategies, we compared full model retraining with parameter–efficient fine–tuning to determine which approach better preserves model performance while ensuring privacy. Full model retraining, though thorough, allows for a comprehensive recovery of information lost during pruning. However, it is computationally expensive and time–consuming, particularly when applying DP, as the added noise complicates the process. In contrast, parameter–efficient fine–tuning, such as LoRA, adjusts only a subset of model parameters, offering a more efficient alternative. LoRA fine–tuning allows for faster adaptation to the pruned model while incorporating privacy safeguards, making it less resource–intensive compared to full retraining. Table 5.8 demonstrates that models fine–tuned with LoRA achieve higher accuracy across various datasets (SST–2, RTE, MRPC, and COLA) compared to those trained without LoRA. For instance, at a sparsity level of 0.1, accuracy on the SST–2 dataset improves from 90.11% to 92.31% with LoRA fine–tuning, with similar improvements observed across other datasets. This trend is consistent across all sparsity levels, suggesting that LoRA fine–tuning is more effective at maintaining or even enhancing model performance after pruning. These findings indicate that LoRA's parameter–efficient fine–tuning outperforms traditional retraining methods in balancing model compression with performance retention.

Table 5.9: Comparison with Baselines

Model	Training	SST–2	QNLI
BERT–base	Finetune	92.31	87.90
BERT–small	Finetune	79.93	81.96
DistilBERT	Finetune	90.90	87.33
1/2–BERT	DPKD [Mir+22]	78.5	80.10
1/2–BERT	Structured DPIMP [Mir+22]	83.3	80.90
SparseBERT	Unstructured DPIMP [Mir+22]	83.7	82.20
BERT–small	*TSKD*	86.23	83.17
BERT–base	**PrunePrivyTune**	92.31	86.51

5.9.6 Comparison with Baselines

We now compare our approach with several baselines such as BERT–base, BERT–small, DistilBERT in Table 5.9. We also provided a fair comparison with other model compression techniques such as our previously proposed *TSKD*, and an existing paper [Mir+22] which is closely related with our work, as it is about knowledge distillation with zero–shot prompting (DPKD) and structured and unstructured pruning (DPIMP). BERT–base having the most parameters i.e., 109M, achieves the highest accuracy across tasks like SST–2 and QNLI, but it is most computationally expensive. In contrast, smaller models like BERT–small, with only 5M parameters, show a significant drop in performance. DistilBERT offers a middle ground with reduced parameters of upto 66M and moderate accuracy. The table also includes approaches like 1/2–BERT and SparseBERT, which use pruning techniques, showing how structured and unstructured pruning can recover some performance while reducing the model size. Differentially Private Knowledge Distillation (DPKD) [Mir+22] resulted in 78.5% accuracy on SST–2 dataset while 80.10% on QNLI which is quite lower than the state–of–the–art BERT models. But authors in [Mir+22] found that pruning performs better than distillation with an accuracy of 83.70% on SST–2 and 82.20% on QNLI dataset. Also, our previous approach *TSKD*, improved the accuracy of the model by 86.23% on SST–2 and 83.17% on QNLI dataset. On the contrary, our approach of *PrunePrivyTune* performs better than the existing works of private model compression and almost similar to the BERT state–of–the–art model which is quite computationally expensive. Finally, the table shows that the proposed approach using LoRA for fine–tuning BERT–base maintains top performance on several datasets matching the full BERT–base fine–tuning results, illustrating the effectiveness of the proposed *PrunePrivyTune* method in preserving performance with efficient parameter usage.

5.9.7 Advantages of Redundancy Based Pruning for DPLoRA

Our pruning method offers significant improvements over traditional techniques like magnitude–based pruning, particularly in the context of DPLoRA. Magnitude–based pruning removes layers based on the magnitude of their weights, assuming that smaller weights contribute less to the model's output. However, this approach fails to consider the semantic redundancy between consecutive layers, which may lead to the pruning of important layers with small weights, resulting in utility loss. In contrast, our redundancy–based approach uses cosine similarity between the hidden states of consecutive layers to identify and prune redundant layers, which contribute minimally to downstream tasks. This selective pruning minimizes utility loss, which is crucial for preserving accuracy during DP fine–tuning.

Furthermore, in DPLoRA, the DP noise depends on gradient sensitivity, which is influenced by the number of parameters and updates in the pruned layers. Traditional pruning methods overlook this aspect, potentially increasing gradient noise variance, leading to inefficiencies. Our approach, by focusing on pruning redundant layers, reduces gradient sensitivity, thereby minimizing the noise injected into the gradients and improving the privacy–utility trade–off.

Additionally, the convergence rate of DPSGD is inversely proportional to noise scale and directly proportional to gradient variance. Traditional pruning methods may inadvertently increase gradient variance, slowing down the convergence process. Our method reduces gradient variance, enabling faster convergence while still adhering to DP constraints, making it more efficient for DPLoRA fine–tuning. Moreover, studies have shown that pruning itself can enhance model privacy [Hua+20], as it helps prevent leakage from membership inference attacks [Wan+20b]. By applying DPLoRA fine–tuning after pruning, we not only improve model privacy but also reduce model storage and computational costs, striking a balance between privacy preservation and resource efficiency.

5.9.8 Quantifying Privacy and Memorization in Synthetic Data

Language models inherently have a tendency to memorize the data they are trained on, which could lead to the leakage of sensitive information. To mitigate this risk and prevent memorization, DP is applied during training. After pruning the model and fine–tuning it with a privacy–preserving approach, we generate the synthetic data with the model to assess its privacy effectiveness, as described in Section 5.7.3. Table 5.10 presents an evaluation of the synthetic data generated across various datasets using two key metrics discussed in Section 5.8.3: Perplexity and BERTScore. Perplexity measures the model's confidence in its predictions. A lower perplexity indicates that the model is more confident in its predictions, suggesting that it may have memorized similar training data. Conversely, higher perplexity suggests that the model is

Table 5.10: Evaluation of Generated Synthetic Data using Perplexity and BERTScore

DP	Metric	SST–2	RTE	MRPC	COLA
No	Perplexity	6.32×10^{5}	5.27×10^{6}	6.29×10^{-3}	4.87×10^{1}
	BERT_Precision	0.3701	0.3931	0.3833	0.3425
	BERT_Recall	0.3698	0.3940	0.3852	0.3481
	BERT_F1	0.3697	0.3935	0.3832	0.3460
ϵ=1	Perplexity	7.10×10^{5}	5.76×10^{6}	1.12×10^{-2}	5.22×10^{4}
	BERT_Precision	0.3536	0.3613	0.3412	0.3384
	BERT_Recall	0.3500	0.3679	0.3484	0.3392
	BERT_F1	0.3512	0.3646	0.3448	0.3380
ϵ=5	Perplexity	6.83×10^{5}	5.52×10^{6}	9.10×10^{-3}	9.25×10^{3}
	BERT_Precision	0.3573	0.3754	0.3650	0.3397
	BERT_Recall	0.3509	0.3848	0.3724	0.3399
	BERT_F1	0.3523	0.3800	0.3678	0.3385
ϵ=10	Perplexity	6.62×10^{5}	5.35×10^{6}	$7.50\times10-3$	1.40×10^{3}
	BERT_Precision	0.3694	0.3879	0.3690	0.3412
	BERT_Recall	0.3690	0.3885	0.3922	0.3474
	BERT_F1	0.3702	0.3882	0.3790	0.3442

less confident and encountering data in a more novel context, which could indicate reduced memorization. BERTScore, which includes Precision, Recall, and F1, evaluates the quality of synthetic text in terms of its similarity to reference text. This metric helps assess how well the generated text resembles human–generated text in terms of semantic similarity, which is important for determining the naturalness and coherence of the synthetic data. These metrics help us determine whether the model is successfully preserving privacy by generating novel and diverse synthetic data, without inadvertently memorizing or leaking sensitive information from the training data.

The results show that as the value of ϵ increases, the model's perplexity generally decreases across all datasets. When $\epsilon = 1$, the model has the highest perplexity, indicating it has limited access to specific details from the training data due to stronger privacy protection. As ϵ increases from 1 to 10, the strength of privacy protection weakens, allowing the model to access and potentially retain more detailed information from the training set. This leads to lower perplexity scores, meaning the model's predictions align more closely with the training data, a sign of increased memorization. Without any differential privacy, the model exhibits the lowest perplexity, reflecting the highest degree of memorization. This trend illustrates the trade–off between privacy and data exposure, stronger privacy guarantees (smaller ϵ) restrict the model from learning precise patterns in the data, while weaker guarantees (larger ϵ) allow the model to memorize more from the training data.

For the BERTScore Precision, Recall, and F1 metrics are observed as ϵ increases, indicating that the synthetic data becomes more similar to the original training data. When ϵ=1, the synthetic data exhibits the least similarity to the training data, reflecting strong privacy preservation through DP. As ϵ increases, privacy protection diminishes, allowing the synthetic data to more closely resemble the original text, which results in higher BERTScore. The highest BERTScore is achieved when no privacy constraints are applied, demonstrating that the synthetic data is most similar to the training data in the absence of privacy guarantees.

5.10 Conclusion

In this chapter, we explored strategies to address two critical challenges associated with language models: high computational overhead and privacy leakage. Due to their massive parameter sizes ranging from millions to billions, these models often suffer from high inference times, limiting their deployment in real–world applications. At the same time, these models are susceptible to privacy risks, as they can inadvertently memorize and reveal the sensitive information from their training data.

To mitigate these concerns and enhance the practical usability of LLMs, we proposed two approaches: *Task-Specific Knowledge Distillation with Differential Privacy* and *PrunePrivyTune*. The first approach leverages transfer learning to perform knowledge distillation from a larger teacher model into a smaller student model, while ensuring privacy guarantees through DP training. This method is especially useful when focusing on a specific downstream task. We demonstrate that the model size can be reduced by up to 95% while preserving the utility, achieving a comparable accuracy to both non-private and private baselines.

To generalize beyond task–specific scenarios, we further studied *pruning* as an effective model compression technique. In particular, we introduced a novel redundancy–based pruning framework that uses pairwise cosine similarity of activation states to identify and remove redundant transformer layers. Layers with high similarity to their neighbors are deemed redundant and pruned, leading to a leaner model with fewer parameters and faster inference. Fine–tuning is essential post–pruning, to recover the lost knowledge. However, when dealing with sensitive data, ensuring privacy during this step is crucial. To this end, we proposed a differentially private version of fine–tuning strategy using LoRA, which we call DPLoRA. Unlike standard LoRA, our method incorporates gradient clipping and Gaussian noise addition to the low–rank matrices, thus providing formal privacy guarantees while maintaining LoRA's efficiency. Our study highlights the synergy between pruning and DPLoRA. Pruning redundant layers reduces the number of parameters needing DP protection, enabling more focused and effective updates. This improves gradient clipping and noise addition in DPSGD, minimizing the utility loss. Additionally, LoRA's

low–rank structure acts as a regularizer, enhancing generalization while preserving privacy. Finally, we evaluated the privacy risks of our method using a training data extraction attack and showed that our approach mitigates the memorization of sensitive data while preserving model utility.

In summary, these strategies serve as a solid foundation for building future scalable, privacy–preserving, and deployable language models.

Chapter 6

Conclusion

> We can only see a short distance ahead, but we can see plenty there that needs to be done
>
> — *Alan Turing*

The work presented in this thesis aims to advance our understanding of developing privacy–aware AI systems. In particular, it explores challenges of existing privacy techniques for high–dimensional data. It also investigates the effectiveness of synthetic data generation as an alternative to anonymization, evaluating its impact on both privacy protection and model utility. Additionally, it extends the investigation to large–scale models, ensuring that privacy constraints do not degrade their performance. A key focus of this work is to achieve an optimal balance between privacy and utility. This chapter summarizes the key findings of this research, highlighting its contributions to advancing privacy–aware machine learning methodologies.

6.1 Reflection on the Research Questions

In this section, we revisit the research questions outlined at the beginning of this thesis and reflect how each has been addressed within the scope of this thesis.

RQ1: **Are existing privacy models and their combinations effective in preserving the privacy and utility of high–dimensional data?**

In Chapter 3, we examined the limitations of traditional privacy techniques such as k–anonymity and DP when applied to high–dimensional data as they often suffer from sparsity and significant utility degradation. Our findings show that k–anonymity struggles with sparsity and DP often reduces utility due to excessive noise. To overcome these limitations,

we explored manifold learning techniques that uncover and preserve the intrinsic structure of the data by projecting it onto a lower–dimensional space. This not only improved the performance of k–anonymity by enabling better clustering but also reduced information loss during anonymization. This approach proved especially effective when the data exhibited a well–defined structure. Building on these insights, we proposed a hybrid privacy framework that combines k–anonymity and DP. This model leverages the grouping efficiency of k–anonymity, the formal guarantees of DP, enhanced through Fréchet mean. Our evaluation showed that this hybrid approach achieves privacy protection comparable to standalone DP while significantly improving data utility. In downstream tasks, it consistently outperformed individual models, offering a better balance between privacy and information retention. We conclude that while traditional privacy models face limitations in high–dimensional settings, thoughtful combinations augmented with manifold learning can provide effective and practical solutions for privacy–preserving machine learning on complex data.

RQ2: **Can synthetic data generation methods capture and preserve the intrinsic manifold structure of high–dimensional data?**

In Chapter 4, we investigated synthetic data generation techniques to assess their ability to capture and preserve the intrinsic manifold structure of high–dimensional data while maintaining privacy. Specifically, we explored whether existing generative models, such as CTGAN, could be adapted for high–dimensional tabular data while ensuring privacy protection, as these models have a tendency to memorize training samples, which can lead to privacy leakage and violate data protection regulations. Through statistical and privacy evaluations, we found that while our proposed framework improves upon existing baselines in some aspects, they also exhibit suboptimal performance in certain cases, highlighting the inability of current methods to fully capture the underlying data correlations. This motivated us to investigate whether incorporating prior knowledge about the data distribution could enhance the quality and privacy of synthetic data generation. We explored various strategies to integrate prior knowledge and found that using Bayesian networks effectively improved synthetic data realism by explicitly modeling dependencies between variables. Beyond evaluating the performance of generative models, we also sought to understand their black–box nature by visualizing latent space representations. Our analysis revealed that for low–dimensional structured datasets, VAEs outperform GANs due to their ability to learn smooth latent representations. However, for more complex high–dimensional datasets, GANs could achieve comparable performance when properly trained. These findings emphasize the importance of enhancing synthetic data generation techniques with domain–specific knowledge to ensure both data utility and privacy pro-

tection, making them more viable for real–world applications.

RQ3: **Can large–scale models like language models leverage privacy models and model compression to ensure privacy and reduce computational overhead?**

In Chapter 5, we addressed the challenges associated with the high dimensionality and privacy concerns of large–scale language models. These models, which contain millions of parameters, often suffer from high inference latency and pose significant privacy risks due to their tendency to memorize sensitive training data. To mitigate these issues, we explored a combination of model compression techniques, such as knowledge distillation and pruning, alongside privacy–preserving methods like differential privacy. Our findings reveal that both approaches effectively reduce model parameters, which in turn lead to decrease in inference time. At the same time, they help protect privacy during training, outperforming existing baselines. However, we observed that each approach performs optimally in different contexts. For scenarios that require a generalized framework and computational efficiency, our pruning method *PrunePrivyTune* is particularly effective, as it not only prunes but also privately fine–tunes the model. On the other hand, when extreme compression is required, especially when pre–trained models are large, the data is complex and multi–tasked, or when a model pre–trained on a similar task is available, then our proposed *TSKD* approach should be preferred. Both frameworks contribute to the improvement of existing model compression techniques, offering a viable solution for deploying large–scale language models in a more cost–efficient and privacy–conscious manner.

6.2 Main Contributions

In summary, the main findings of this thesis are outlined below. Each contribution corresponds to one of the research papers included in this work.

- We investigated the use of manifold learning techniques in combination with the k–anonymity privacy model to preserve the intrinsic structure of high–dimensional data while anonymizing it, ensuring privacy while maintaining utility. **(Paper I)**
- We proposed a hybrid anonymization model that integrates the strengths of k–anonymity and differential privacy. Additionally, we analyzed the trade–off between the privacy parameters (ϵ of DP and k of k–anonymity) in terms of utility preservation. **(Paper II)**
- We designed a privacy–preserving synthetic data generation framework that maintains the manifold properties of the original data while mitigating privacy risks, and assess the effectiveness of our approach through data reconstruction attacks. **(Paper III)**

- We investigated the distribution learning capabilities of generative models to better understand their black–box nature and latent space representations. Our analysis showed that VAE excels at capturing low–dimensional point distributions, offering insights into their ability to model structured data. **(Paper IV)**

- We explored strategies to incorporate prior knowledge into GANs to improve the quality of synthetic data. We found that Bayesian networks can effectively capture attribute dependencies and serve as a structured prior for training GANs. **(Paper V)**

- We studied whether large–scale language models can be efficiently compressed to reduce inference time while ensuring privacy–preserving training to prevent memorization of sensitive data. Our task–specific knowledge distillation approach compressed model parameters by 95%, achieving performance comparable to non–private training. **(Paper VI)**

- We further investigated pruning–based model compression and also proposed a method to recover the lost information using differentially private fine–tuning. To assess privacy vulnerabilities, we conducted training data extraction attack, demonstrating that our approach provides enhanced privacy protection while maintaining model efficiency. **(Paper VII)**

6.3 Future Work

One interesting avenue for future research is deepening our understanding of the proposed hybrid anonymization technique that synergizes k–anonymity and differential privacy. This could be studied by developing dynamic methods that adjust privacy parameters (k and ϵ) based on the sensitivity of different data regions, as traditional privacy techniques overlook distinct sensitivities. Such an approach would mitigate excessive data distortion while ensuring robust privacy protection. Moreover, adapting this hybrid privacy technique to diverse data modalities, such as sequential data and graph–structured data, is another important direction. For sequential data, it would be essential to preserve temporal dependencies while ensuring strong privacy guarantees. Similarly, in graph-based applications (e.g., social networks or knowledge graphs), anonymizing node–link structures without losing relational information remains a critical challenge. Another crucial research direction is evaluating the resilience of this privacy model against different types of privacy attacks. Developing adaptive countermeasures against these threats could significantly strengthen the real–world applicability of the hybrid anonymization framework.

Another interesting future direction is the development of interpretable synthetic data generation models. While several techniques exist for generating synthetic data, assessing the quality and validity of both the synthetic data and the underlying generative models remains a significant challenge. Current

methods often lack transparency, making it difficult to evaluate how well the synthetic data captures the real–world distribution or whether privacy guarantees are maintained without compromising data utility. The goal is to enhance the interpretability of both the data generation process and the properties of the synthetic data, enabling more effective evaluation of its quality and ensuring that privacy protections are preserved.

Further research could explore the development of foundation models with built–in forgetting mechanisms, enabling models to selectively forget specific data after training. This capability is essential for compliance with privacy regulations, such as GDPR, which require models to remove sensitive or personal information upon request. Current models lack the ability to efficiently unlearn specific data without requiring costly retraining from scratch. By designing modular architectures and efficient unlearning algorithms, we can isolate the influence of sensitive data within the model and remove it without sacrificing overall model performance. This approach will enhance the privacy compliance and adaptability of LLMs, making them more suitable for deployment in regulated industries, such as healthcare and finance, where privacy concerns are paramount.

This thesis provides a foundation for advancing privacy–aware AI systems by addressing key challenges across multiple dimensions. We propose novel solutions for protecting high–dimensional data through hybrid privacy models, generate high–quality synthetic data that balance utility and privacy, and develop strategies to mitigate privacy risks and computational overhead in language models. These contributions aim to enhance the trustworthiness and applicability of machine learning systems in privacy–sensitive domains. The outlined future directions present promising avenues to extend this work further, enabling the continued development of secure and interpretable AI solutions in an increasingly data–driven world.

Bibliography

[Aba+16] Martin Abadi et al. "Deep learning with differential privacy". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318.

[Agg05] Charu C Aggarwal. "On k-anonymity and the curse of dimensionality". In: *VLDB*. Vol. 5. 2005, pp. 901–909.

[Agg06] Charu C Aggarwal. "On randomization, public information and the curse of dimensionality". In: *2007 IEEE 23rd International Conference on Data Engineering*. IEEE. 2006, pp. 136–145.

[Akr+20] Haleh Akrami et al. "Robust variational autoencoder for tabular data with beta divergence". In: *arXiv preprint arXiv:2006.08204* (2020).

[AP15] Ankur Ankan and Abinash Panda. "pgmpy: Probabilistic Graphical Models using Python". In: *Proceedings of the Python in Science Conference*. SciPy. SciPy, 2015. URL: `http://dx.doi.org/10.25080/Majora-7b98e3ed-001`.

[AP23] Kamil Adamczewski and Mijung Park. "Differential privacy meets neural network pruning". In: *arXiv preprint arXiv:2303.04612* (2023).

[Ata+99] Mike Atallah et al. "Disclosure limitation of sensitive rules". In: *Proceedings 1999 Workshop on Knowledge and Data Engineering Exchange (KDEX'99)(Cat. No. PR00453)*. IEEE. 1999, pp. 45–52.

[Atz+08] Maurizio Atzori et al. "Anonymity preserving pattern discovery". In: *The VLDB journal* 17 (2008), pp. 703–727.

[AW10] Hervé Abdi and Lynne J Williams. "Principal component analysis". In: *Wiley interdisciplinary reviews: computational statistics* 2.4 (2010), pp. 433–459.

[AY04] Charu C Aggarwal and Philip S Yu. "A condensation approach to privacy preserving data mining". In: *Advances in Database Technology-EDBT 2004: 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004 9*. Springer. 2004, pp. 183–199.

[BCN06] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. "Model compression". In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2006, pp. 535–541.

[BDS18] Andrew Brock, Jeff Donahue, and Karen Simonyan. "Large scale GAN training for high fidelity natural image synthesis". In: *arXiv preprint arXiv:1809.11096* (2018).

[Ber13] Wicher Bergsma. "A bias-correction for Cramér's V and Tschuprow's T". In: *Journal of the Korean Statistical Society* 42.3 (2013), pp. 323–328.

[BK96] Barry Becker and Ronny Kohavi. *Adult*. UCI Machine Learning Repository. Accessed on: March 2025. 1996.

[Bra02] Ruth Brand. "Microdata protection through noise addition". In: *Inference Control in Statistical Databases: From Theory to Practice* (2002), pp. 97–116.

[Bre+14] Robert Bredereck et al. "The effect of homogeneity on the computational complexity of combinatorial data anonymization". In: *Data Mining and Knowledge Discovery* 28 (2014), pp. 65–91.

[Bro20] Tom B Brown. "Language models are few-shot learners". In: *arXiv preprint ArXiv:2005.14165* (2020).

[BT13] Johan Barthelemy and Philippe L Toint. "Synthetic population generation without a sample". In: *Transportation Science* 47.2 (2013), pp. 266–279.

[Car+19] Nicholas Carlini et al. "The secret sharer: Evaluating and testing unintended memorization in neural networks". In: *28th USENIX security symposium (USENIX security 19)*. 2019, pp. 267–284.

[Car+21] Nicholas Carlini et al. "Extracting training data from large language models". In: *30th USENIX Security Symposium (USENIX Security 21)*. 2021, pp. 2633–2650.

[Cay+08] Lawrence Cayton et al. *Algorithms for manifold learning*. eScholarship, University of California, 2008.

[CH92] Gregory F Cooper and Edward Herskovits. "A Bayesian method for the induction of probabilistic networks from data". In: *Machine learning* 9 (1992), pp. 309–347.

[Che+05] Hongwei Cheng et al. “On the compression of low rank matrices”. In: *SIAM Journal on Scientific Computing* 26.4 (2005), pp. 1389–1404.

[Che+17] Tong Che et al. “Maximum-likelihood augmented discrete generative adversarial networks”. In: *arXiv preprint arXiv:1702.07983* (2017).

[Che+24] Xiaodong Chen et al. “Streamlining redundant layers to compress large language models”. In: *arXiv preprint arXiv:2403.19135* (2024).

[Cho+17] Edward Choi et al. “Generating multi-label discrete patient records using generative adversarial networks”. In: *Machine learning for healthcare conference*. PMLR. 2017, pp. 286–305.

[Cho+23] Aakanksha Chowdhery et al. “Palm: Scaling language modeling with pathways”. In: *Journal of Machine Learning Research* 24.240 (2023), pp. 1–113.

[CKY23] Zhoujun Cheng, Jungo Kasai, and Tao Yu. “Batch prompting: Efficient inference with large language model apis”. In: *arXiv preprint arXiv:2301.08721* (2023).

[CN18] Maximin Coavoux and Shashi Narayan. “Privacy-preserving neural representations of text”. In: *arXiv preprint arXiv:1808.09408* (2018).

[Coh+09] Israel Cohen et al. “Pearson correlation coefficient”. In: *Noise reduction in speech processing* (2009), pp. 1–4.

[CS14] Girish Chandrashekar and Ferat Sahin. “A survey on feature selection methods”. In: *Computers & electrical engineering* 40.1 (2014), pp. 16–28.

[Dal+20] Fahim Dalvi et al. “Analyzing redundancy in pretrained transformer models”. In: *arXiv preprint arXiv:2004.04010* (2020).

[Dat23] Inc. DataCebo. *Synthetic Data Metrics*. Accessed on: Mar 2025. 2023. URL: `https://docs.sdv.dev/sdmetrics/`.

[De +12] Sabrina De Capitani Di Vimercati et al. “Data privacy: Definitions and techniques”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 20.06 (2012), pp. 793–817.

[Den+09] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[Dev18] Jacob Devlin. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[Dij22] Edsger W Dijkstra. “A note on two problems in connexion with graphs”. In: *Edsger Wybe Dijkstra: his life, work, and legacy*. 2022, pp. 287–290.

[DM02] Josep Domingo-Ferrer and Josep Maria Mateo-Sanz. “Practical data-oriented microaggregation for statistical disclosure control”. In: *IEEE Transactions on Knowledge and data Engineering* 14.1 (2002), pp. 189–201.

[Dok+15] Ivan Dokmanic et al. “Euclidean distance matrices: essential theory, algorithms, and applications”. In: *IEEE Signal Processing Magazine* 32.6 (2015), pp. 12–30.

[Dom+06] Josep Domingo-Ferrer et al. “Efficient multivariate data-oriented microaggregation”. In: *The VLDB Journal* 15 (2006), pp. 355–369.

[Dom07] Josep Domingo-Ferrer. “A three-dimensional conceptual framework for database privacy”. In: *Secure Data Management: 4th VLDB Workshop, SDM 2007, Vienna, Austria, September 23-24, 2007. Proceedings 4*. Springer. 2007, pp. 193–202.

[Dom08] Josep Domingo-Ferrer. “A survey of inference control methods for privacy-preserving data mining”. In: *Privacy-Preserving Data Mining: Models and Algorithms* (2008), pp. 53–80.

[DR82] Tore Dalenius and Steven P Reiss. “Data-swapping: A technique for disclosure control”. In: *Journal of statistical planning and inference* 6.1 (1982), pp. 73–85.

[Dre11] Jörg Drechsler. *Synthetic datasets for statistical disclosure control: theory and implementation*. Vol. 201. Springer Science & Business Media, 2011.

[DT01a] Josep Domingo-Ferrer and Vicenc Torra. “A quantitative comparison of disclosure control methods for microdata”. In: *Confidentiality, disclosure and data access: theory and practical applications for statistical agencies* (2001), pp. 111–134.

[DT01b] Josep Domingo-Ferrer and Vicenc Torra. “Disclosure control methods and information loss for microdata”. In: *Confidentiality, disclosure, and data access: theory and practical applications for statistical agencies* (2001), pp. 91–110.

[DT05] Josep Domingo-Ferrer and Vicenç Torra. “Ordinal, continuous and heterogeneous k-anonymity through microaggregation”. In: *Data Mining and Knowledge Discovery* 11 (2005), pp. 195–212.

[Dwo06] Cynthia Dwork. “Differential privacy”. In: *International colloquium on automata, languages, and programming*. Springer. 2006, pp. 1–12.

[FA23] Elias Frantar and Dan Alistarh. “Sparsegpt: Massive language models can be accurately pruned in one-shot”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 10323–10337.

[Fio16] Samuele Fiorini. *gene expression cancer RNA-Seq*. UCI Machine Learning Repository. Accessed on: Mar 2025. 2016.

[Fis36] Ronald A Fisher. “The use of multiple measurements in taxonomic problems”. In: *Annals of eugenics* 7.2 (1936), pp. 179–188.

[FJR15] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. “Model inversion attacks that exploit confidence information and basic countermeasures”. In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015, pp. 1322–1333.

[Flo62] Robert W Floyd. “Algorithm 97: shortest path”. In: *Communications of the ACM* 5.6 (1962), pp. 345–345.

[FM53] Robert Fortet and Edith Mourier. “Convergence de la répartition empirique vers la répartition théorique”. In: *Annales scientifiques de l’École Normale Supérieure*. Vol. 70. 3. 1953, pp. 267–285.

[Fré] Maurice Fréchet. *Fréchet mean*. Accessed on: Mar 2025. URL: `https://en.wikipedia.org/wiki/Fr%C3%A9chet_mean`.

[Fun+11] Benjamin CM Fung et al. “Service-oriented architecture for high-dimensional private data mashup”. In: *IEEE Transactions on Services Computing* 5.3 (2011), pp. 373–386.

[GDP18] GDPR-Info.eu. *General Data Protection Regulation (GDPR) – The Complete Guide*. Accessed on: Mar 2025. 2018.

[Gha+19] Marjan Ghazvininejad et al. “Mask-predict: Parallel decoding of conditional masked language models”. In: *arXiv:1904.09324* (2019).

[GK73] Karsten Grove and Hermann Karcher. “How to conjugate C 1-close group actions”. In: *Mathematische Zeitschrift* 132.1 (1973), pp. 11–20.

[Gon+14] Yunchao Gong et al. “Compressing deep convolutional networks using vector quantization”. In: *arXiv preprint arXiv:1412.6115* (2014).

[Goo+20] Ian Goodfellow et al. “Generative adversarial networks”. In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[Gre+12] Arthur Gretton et al. “A kernel two-sample test”. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 723–773.

[GTK08] Gabriel Ghinita, Yufei Tao, and Panos Kalnis. “On the anonymization of sparse high-dimensional data”. In: *2008 IEEE 24th International Conference on Data Engineering*. IEEE. 2008, pp. 715–724.

[Guy+04] Isabelle Guyon et al. *Gisette*. UCI Machine Learning Repository. Accessed on: Mar 2025. 2004.

[Guy04] Isabelle Guyon. *Madelon*. UCI Machine Learning Repository. Accessed on: March 2025. 2004.

[Han+15] Song Han et al. "Learning both weights and connections for efficient neural network". In: *Advances in neural information processing systems* 28 (2015).

[HAP17] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. "Deep models under the GAN: information leakage from collaborative deep learning". In: *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 2017, pp. 603–618.

[Hay+17] Jamie Hayes et al. "Logan: Membership inference attacks against generative models". In: *arXiv preprint arXiv:1705.07663* (2017).

[Heu+17] Martin Heusel et al. "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *Advances in neural information processing systems* 30 (2017).

[Hev+04] Alan R Hevner et al. "Design science in information systems research". In: *MIS quarterly* (2004), pp. 75–105.

[HHS96] HHS-US. *Health Insurance Portability and Accountability Act*. Accessed on: Mar 2025. 1996.

[HN09] Yeye He and Jeffrey F Naughton. "Anonymization of set-valued data via top-down, local generalization". In: *Proceedings of the VLDB Endowment* 2.1 (2009), pp. 934–945.

[Hof94] Hans Hofmann. "Statlog (German Credit Data)". In: (1994). Accessed on: March 2025.

[Hop02] Mark Hopkins. *https://archive.ics.uci.edu/ml/datasets/ spambase*. Accessed on: Mar 2025. 2002.

[Hot33] Harold Hotelling. "Analysis of a complex of statistical variables into principal components." In: *Journal of educational psychology* 24.6 (1933), p. 417.

[Hou+20] Lu Hou et al. "Dynabert: Dynamic bert with adaptive width and depth". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9782–9793.

[HP88] Stephen Hanson and Lorien Pratt. "Comparing biases for minimal network construction with back-propagation". In: *Advances in neural information processing systems* 1 (1988).

[Hu+21] Edward J Hu et al. "Lora: Low-rank adaptation of large language models". In: *arXiv preprint arXiv:2106.09685* (2021).

[Hua+20] Yangsibo Huang et al. "Privacy-preserving learning via deep net pruning". In: *arXiv preprint arXiv:2003.01876* (2020).

[Hun+12] Anco Hundepool et al. *Statistical disclosure control*. John Wiley & Sons, 2012.

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network". In: *arXiv preprint arXiv:1503.02531* (2015).

[HW01] Zengyi Huang and Paul Williamson. "A comparison of synthetic reconstruction and combinatorial optimisation approaches to the creation of small-area microdata". In: *Department of Geography, University of Liverpool* (2001).

[HX23] Yang He and Lingao Xiao. "Structured pruning for deep convolutional neural networks: A survey". In: *IEEE transactions on pattern analysis and machine intelligence* 46.5 (2023), pp. 2900–2919.

[HYW18] He Huang, Philip S Yu, and Changhu Wang. "An introduction to image synthesis with generative adversarial nets". In: *arXiv preprint arXiv:1803.04469* (2018).

[Jol02] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.

[Jos+18] Matthew Joseph et al. "Local differential privacy for evolving data". In: *Advances in Neural Information Processing Systems* 31 (2018).

[JT09] Iain M Johnstone and D Michael Titterington. *Statistical challenges of high-dimensional data*. 2009.

[JT11] Czapinski J. and Panek T. *SD2011*. http://www.diagnoza.com/index-en.html. Accessed on: March 2025. 2011.

[Kag] Kaggle. *Breast Cancer Dataset*. Accessed on: March 2025.

[Kam+19] Gautam Kamath et al. "Privately learning high-dimensional distributions". In: *Conference on Learning Theory*. PMLR. 2019, pp. 1853–1902.

[Kar+05] Hillol Kargupta et al. "Random-data perturbation techniques and privacy-preserving data mining". In: *Knowledge and Information Systems* 7.4 (2005), pp. 387–414.

[Kar+17] Tero Karras et al. "Progressive growing of gans for improved quality, stability, and variation". In: *arXiv preprint arXiv:1710.10196* (2017).

[KB07] Markus Kalisch and Peter Bühlman. "Estimating high-dimensional directed acyclic graphs with the PC-algorithm." In: *Journal of Machine Learning Research* 8.3 (2007).

[KH16] Matt J Kusner and José Miguel Hernández-Lobato. "Gans for sequences of discrete elements with the gumbel-softmax distribution". In: *arXiv preprint arXiv:1611.04051* (2016).

[Kin13] Diederik P Kingma. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[Kru64] Joseph B Kruskal. “Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis”. In: *Psychometrika* 29.1 (1964), pp. 1–27.

[Kur+22] Eldar Kurtic et al. “The optimal bert surgeon: Scalable and accurate second-order pruning for large language models”. In: *arXiv preprint arXiv:2203.07259* (2022).

[Kwo+20] Se Jung Kwon et al. “Structured compression by weight encryption for unstructured pruning and quantization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 1909–1918.

[LBC18] Yitong Li, Timothy Baldwin, and Trevor Cohn. “Towards robust and privacy-preserving text representations”. In: *arXiv preprint arXiv:1805.06093* (2018).

[Lee+20] Jaeho Lee et al. “Layer-adaptive sparsity for the magnitude-based pruning”. In: *arXiv preprint arXiv:2010.07611* (2020).

[Lee18] John M Lee. *Introduction to Riemannian manifolds*. Vol. 2. Springer, 2018.

[LGS13] Grigorios Loukides, Aris Gkoulalas-Divanis, and Jianhua Shao. “Efficient and flexible anonymization of transaction data”. In: *Knowledge and information systems* 36 (2013), pp. 153–210.

[Li+19] Wanjie Li et al. “PPDP-PCAO: an efficient high-dimensional data releasing method with differential privacy protection”. In: *IEEE access* 7 (2019), pp. 176429–176437.

[Li+21] Xuechen Li et al. “Large language models can be strong differentially private learners”. In: *arXiv preprint arXiv:2110.05679* (2021).

[LLV06] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. “t-closeness: Privacy beyond k-anonymity and l-diversity”. In: *2007 IEEE 23rd international conference on data engineering*. IEEE. 2006, pp. 106–115.

[LWZ08] Lian Liu, Jie Wang, and Jun Zhang. “Wavelet-based data perturbation for simultaneous privacy-preserving and statistics-preserving”. In: *2008 IEEE International Conference on Data Mining Workshops*. IEEE. 2008, pp. 27–35.

[Mac+07] Ashwin Machanavajjhala et al. “l-diversity: Privacy beyond k-anonymity”. In: *Acm transactions on knowledge discovery from data (tkdd)* 1.1 (2007), 3–es.

[Man11] S Manikandan. “Measures of central tendency: The mean”. In: *Journal of Pharmacology and Pharmacotherapeutics* 2.2 (2011), p. 140.

[Mar11] Stephen Marsland. *Machine learning: an algorithmic perspective*. Chapman and Hall/CRC, 2011.

[Meh+22] Harsh Mehta et al. “Large scale transfer learning for differentially private image classification”. In: *arXiv preprint arXiv:2205.02973* (2022).

[MFW23] Xinyin Ma, Gongfan Fang, and Xinchao Wang. “Llm-pruner: On the structural pruning of large language models”. In: *Advances in neural information processing systems* 36 (2023), pp. 21702–21720.

[MHM18] Leland McInnes, John Healy, and James Melville. “Umap: Uniform manifold approximation and projection for dimension reduction”. In: *arXiv preprint arXiv:1802.03426* (2018).

[Mik+13] Tomas Mikolov et al. “Distributed representations of words and phrases and their compositionality”. In: *Advances in neural information processing systems* 26 (2013).

[Mir+20] Fatemehsadat Mireshghallah et al. “Privacy in deep learning: A survey”. In: *arXiv preprint arXiv:2004.12254* (2020).

[Mir+22] Fatemehsadat Mireshghallah et al. “Differentially private model compression”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 29468–29483.

[Mir14] Mehdi Mirza. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014).

[Moo96] RA Moore. “Controlled data swapping for masking public use microdata sets”. In: *US Census Bureau Research Report* 96.04 (1996).

[MS05] Krishnamurty Muralidhar and Rathindra Sarathy. “An enhanced data perturbation approach for small data sets”. In: *Decision Sciences* 36.3 (2005), pp. 513–529.

[MS24] Marko Miletic and Murat Sariyar. “Challenges of Using Synthetic Data Generation Methods for Tabular Microdata”. In: *Applied Sciences* 14.14 (2024), p. 5975.

[Naj+15] Maryam M Najafabadi et al. “Deep learning applications and challenges in big data analytics”. In: *Journal of big data* 2 (2015), pp. 1–21.

[NK17] Maximillian Nickel and Douwe Kiela. “Poincaré embeddings for learning hierarchical representations”. In: *Advances in neural information processing systems* 30 (2017).

[OOS17] Augustus Odena, Christopher Olah, and Jonathon Shlens. "Conditional image synthesis with auxiliary classifier gans". In: *International conference on machine learning*. PMLR. 2017, pp. 2642–2651.

[Pap+16] Nicolas Papernot et al. "Semi-supervised knowledge transfer for deep learning from private training data". In: *arXiv preprint:1610.05755* (2016).

[Pap+18] Nicolas Papernot et al. "Scalable private learning with pate". In: *arXiv preprint arXiv:1802.08908* (2018).

[Par+18] Noseong Park et al. "Data synthesis based on generative adversarial networks". In: *arXiv preprint arXiv:1806.03384* (2018).

[Par18] Stuart L Pardau. "The california consumer privacy act: Towards a european-style privacy regime in the united states". In: *J. Tech. L. & Pol'y* 23 (2018), p. 68.

[Ped+11] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[Pon+23] Natalia Ponomareva et al. "How to dp-fy ml: A practical guide to machine learning with differential privacy". In: *Journal of Artificial Intelligence Research* 77 (2023), pp. 1113–1201.

[PY09] Sinno Jialin Pan and Qiang Yang. "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.

[RBS21] Matthew Reimherr, Karthik Bharath, and Carlos Soto. "Differential privacy over Riemannian manifolds". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 12292–12303.

[RM19] Amit Rege and Claire Monteleoni. "Evaluating the distribution learning capabilities of GANs". In: *arXiv preprint arXiv:1907.02662* (2019).

[RMC15] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[RR20] Mayur Rathi and Anand Rajavat. "High Dimensional Data Processing in Privacy Preserving Data Mining". In: *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*. IEEE. 2020, pp. 212–217.

[RS00] Sam T Roweis and Lawrence K Saul. "Nonlinear dimensionality reduction by locally linear embedding". In: *science* 290.5500 (2000), pp. 2323–2326.

[Rub93] Donald B Rubin. "Statistical disclosure limitation". In: *Journal of official Statistics* 9.2 (1993), pp. 461–468.

[Sam01] Pierangela Samarati. “Protecting respondents identities in microdata release”. In: *IEEE transactions on Knowledge and Data Engineering* 13.6 (2001), pp. 1010–1027.

[San+19] Victor Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *arXiv preprint arXiv:1910.01108* (2019).

[SD17] Jordi Soria-Comas and Josep Domingo-Ferrer. “Differentially private data sets based on microaggregation and record perturbation”. In: *Modeling Decisions for Artificial Intelligence: 14th International Conference, MDAI 2017, Kitakyushu, Japan, October 18-20, 2017, Proceedings 14*. Springer. 2017, pp. 119–131.

[Sho+17] Reza Shokri et al. “Membership inference attacks against machine learning models”. In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[SMD06] Agusti Solanas, Antoni Martínez-Ballesté, and J Domingo-Ferrer. “V-MDAV: a multivariate microaggregation with variable group size”. In: *17th COMPSTAT Symposium of the IASC, Rome*. 2006, pp. 917–925.

[Sná+17] Václav Snášel et al. “Geometrical and topological approaches to Big Data”. In: *Future Generation Computer Systems* 67 (2017), pp. 286–296.

[SO17] Jun Sakuma and Tatsuya Osame. “Recommendation with k-anonymized ratings”. In: *arXiv preprint arXiv:1707.03334* (2017).

[Spr14] Vincent Spruyt. “The curse of dimensionality in classification”. In: *Computer vision for dummies* 21.3 (2014), pp. 35–40.

[SS98] Pierangela Samarati and Latanya Sweeney. “Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression”. In: *technical report, SRI International* (1998).

[SSM98] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. “Nonlinear component analysis as a kernel eigenvalue problem”. In: *Neural computation* 10.5 (1998), pp. 1299–1319.

[STV04] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Kernel methods in computational biology*. MIT press, 2004.

[Tao+21] Yuchao Tao et al. “Benchmarking differentially private synthetic data generation algorithms”. In: *arXiv preprint arXiv:2112.09238* (2021).

[TBA06] Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. “The max-min hill-climbing Bayesian network structure learning algorithm”. In: *Machine learning* 65 (2006), pp. 31–78.

[Ten] Tensorflow. *MNIST Dataset*. Accessed on: March 2025.

[TMK08] Manolis Terrovitis, Nikos Mamoulis, and Panos Kalnis. "Privacy-preserving anonymization of set-valued data". In: *Proceedings of the VLDB Endowment* 1.1 (2008), pp. 115–125.

[Tor04] Vicenç Torra. "Microaggregation for categorical variables: a median based approach". In: *Privacy in Statistical Databases: CASC Project Final Conference, PSD 2004, Barcelona, Spain, June 9-11, 2004. Proceedings*. Springer. 2004, pp. 162–174.

[Tor17] Vicenç Torra. *Data privacy: foundations, new developments and the big data challenge*. Vol. 28. Springer, 2017.

[Tor22] Vicenç Torra. *A Guide to Data Privacy*. Springer, 2022.

[TSL00] Joshua B Tenenbaum, Vin de Silva, and John C Langford. "A global geometric framework for nonlinear dimensionality reduction". In: *science* 290.5500 (2000), pp. 2319–2323.

[Tur+19] Iulia Turc et al. "Well-Read Students Learn Better: On the Importance of Pre-training Compact Models". In: *arXiv preprint arXiv:1908.08962v2* (2019).

[Ull17] J. Ullman. *(CS7880): Rigorous approaches to Data Privacy*. Accessed on: March 2025. 2017.

[Uni48] United Nations. *Universal Declaration of Human Rights, Article 12*. Accessed on: Mar 2025. 1948. URL: `https://www.un.org/en/about-us/universal-declaration-of-human-rights`.

[Vak+22] Thomas Vakili et al. "Downstream task performance of BERT models pre-trained using automatically de-identified clinical data". In: *Proceedings of the thirteenth language resources and evaluation conference*. 2022, pp. 4245–4252.

[Var+12] Nebu Varghese et al. "A survey of dimensionality reduction and classification methods". In: *International Journal of Computer Science and Engineering Survey* 3.3 (2012), p. 45.

[VC04] Jaideep Vaidya and Chris Clifton. "Privacy-preserving data mining: Why, how, and when". In: *IEEE Security & Privacy* 2.6 (2004), pp. 19–27.

[VD22] Thomas Vakili and Hercules Dalianis. "Utility preservation of clinical text after De-Identification". In: *Proceedings of the 21st Workshop on Biomedical Language Processing*. 2022, pp. 383–388.

[VH08] Laurens Van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE." In: *Journal of machine learning research* 9.11 (2008).

[Wan+18] Alex Wang et al. "GLUE: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461* (2018).

[Wan+20a] Rong Wang et al. “Privacy-preserving high-dimensional data publishing for classification”. In: *Computers & Security* 93 (2020), p. 101785.

[Wan+20b] Yijue Wang et al. “Against membership inference attack: Pruning is all you need”. In: *arXiv preprint arXiv:2008.13578* (2020).

[Web+19] Ryan Webster et al. “Detecting overfitting of deep generative networks via latent recovery”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 11273–11282.

[Wu+16] Jiaxiang Wu et al. “Quantized convolutional neural networks for mobile devices”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4820–4828.

[WWL19] Ziheng Wang, Jeremy Wohlwend, and Tao Lei. “Structured pruning of large language models”. In: *arXiv preprint arXiv:1910.04732* (2019).

[Xie+18] Liyang Xie et al. “Differentially private generative adversarial network”. In: *arXiv preprint arXiv:1802.06739* (2018).

[Xu+19] Lei Xu et al. “Modeling tabular data using conditional gan”. In: *Advances in neural information processing systems* 32 (2019).

[Xu+23] Zheng Xu et al. “Federated learning of gboard language models with differential privacy”. In: *arXiv preprint arXiv:2305.18465* (2023).

[XV18] Lei Xu and Kalyan Veeramachaneni. “Synthesizing tabular data using generative adversarial networks”. In: *arXiv preprint:1811.11264* (2018).

[YDV20] Jinsung Yoon, Lydia N Drumright, and Mihaela Van Der Schaar. “Anonymization through data synthesis using generative adversarial networks (ads-gan)”. In: *IEEE journal of biomedical and health informatics* 24.8 (2020), pp. 2378–2388.

[Yeo+21] Seul-Ki Yeom et al. “Pruning by explaining: A novel criterion for deep neural network pruning”. In: *Pattern Recognition* 115 (2021), p. 107899.

[YJS19] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. “PATE-GAN: Generating synthetic data with differential privacy guarantees”. In: *International Conference on Learning Representations*. Vol. 2. OpenReview, 2019.

[Yu+17] Lantao Yu et al. “Sequence generative adversarial nets with policy gradient. 492 In”. In: *AAAI conference on artificial intelligence*. Vol. 493. 2017.

[ZG17] Michael Zhu and Suyog Gupta. “To prune, or not to prune: exploring the efficacy of pruning for model compression”. In: *arXiv preprint arXiv:1710.01878* (2017).

[Zha+20] Tianyi Zhang* et al. “BERTScore: Evaluating Text Generation with BERT”. In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=SkeHuCVFDr.

[Zha+21] Zilong Zhao et al. “Ctab-gan: Effective table data synthesizing”. In: *Asian Conference on Machine Learning*. PMLR. 2021, pp. 97–112.

[Zhu+17] Tianqing Zhu et al. “Differentially private data publishing and analysis: A survey”. In: *IEEE Transactions on Knowledge and Data Engineering* 29.8 (2017), pp. 1619–1638.